1991-06

The error performance analysis over cyclic redundancy check codes.

Yoon, Hee Byung.

Monterey, California. Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

THE ERROR PERFORMANCE ANALYSIS OVER
CYCLIC REDUNDANCY CHECK CODES

by

Yoon, Hee Byung

June 1991

Thesis Advisor: Chyan Yang

Approved for public release; distribution is unlimited

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution is unlimited
2b DECLASSIFICATION / DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S)
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School 6b OFFICE SYMBOL (If applicable) EC	7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School
6c ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000	7b ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000
8a NAME OF FUNDING / SPONSORING ORGANIZATION 8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c ADDRESS (City, State, and ZIP Code)	10 SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO / PROJECT NO / TASK NO / WORK UNIT ACCESSION NO

11 TITLE (Include Security Classification) THE ERROR PERFORMANCE ANALYSIS OVER CYCLIC REDUNDANCY CHECK CODES

12 PERSONAL AUTHOR(S)
YOON, Hee Byung

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Master's Thesis	FROM _____ TO _____	June 1991	77

16 SUPPLEMENTARY NOTATION The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the US Government.

17 COSATI CODES		
FIELD	GROUP	SUB-GROUP

18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
Error Performance Analysis; Cyclic Redundancy Check Codes; CRC Block Burst Error

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

The burst error is generated in digital communication networks by various unpredictable conditions, which occur at high error rates, for short durations, and can impact services. To completely describe a burst error one has to know the bit pattern. This is impossible in practice on working systems. Therefore, under the memoryless binary symmetric channel (MBSC) assumptions, the performance evaluation or estimation schemes for digital signal (DS) 1 transmission systems carrying live traffic is an interesting and important problem.

This study will present some analytical methods, leading to efficient detecting algorithms of burst error using cyclic redundancy check (CRC) code. The definition of burst error is introduced using three different models. Among the three burst error models, the mathematical model is

20 DISTRIBUTION / AVAILABILITY OF ABSTRACT	21 ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	UNCLASSIFIED
22a NAME OF RESPONSIBLE INDIVIDUAL YANG, Chyan	22b TELEPHONE (Include Area Code) 408-646-2266 22c OFFICE SYMBOL EC/Ya

DD Form 1473, JUN 86 *Previous editions are obsolete.*

S/N 0102-LF-014-6603

19. cont.
used in this study.
The probability density function f(b) of burst error of length b is proposed. The performance of CRC-n codes is evaluated and analyzed using f(b) through the use of computer simulation model within CRC block burst error. The simulation result shows that the mean block burst error tends to approach the pattern of the burst error which random bit errors generate.

Approved for public release; distribution is unlimited

The Error Performance Analysis over Cyclic Redundancy Check Codes

by

Yoon, Hee Byung
LT, Korean Navy
B.S, Naval Academy Korea, 1983

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL AND COMPUTER ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL

June, 1991

ABSTRACT

The burst error is generated in digital communication networks by various unpredictable conditions, which occur at high error rates, for short durations, and can impact services. To completely describe a burst error one has to know the bit pattern. This is impossible in practice on working systems. Therefore, under the memoryless binary symmetric channel (MBSC) assumptions, the performance evaluation or estimation schemes for digital signal 1 (DS1) transmission systems carrying live traffic is an interesting and important problem.

This study will present some analytical methods, leading to efficient detecting algorithms of burst error using cyclic redundancy check (CRC) code. The definition of burst error is introduced using three different models. Among the three burst error models, the mathematical model is used in this study.

The probability density function $f(b)$ of burst error of length b is proposed. The performance of CRC-n codes is evaluated and analyzed using $f(b)$ through the use of a computer simulation model within CRC block burst error. The simulation result shows that the mean block burst error tends to approach the pattern of the burst error which random bit errors generate.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENT

I would like to thank the Korean Navy for the opportunity to study at the Naval Postgraduate School. I wish to thank Dr. Chyan Yang for his patient guidance, continuous assistance and very helpful criticism throughout this work. I am very grateful to Dr. Tri T. Ha whose comments and recommendations contributed to the successful completion of this thesis. Throughout this thesis preparation, Dr. Jai Eu has helped in confirming my understanding of the subject, revising my approach to the simulation, and best of all giving me the confidence of doing useful research work that is of industrial interest. Finally, I am grateful to my mother and also grateful to my wife, Kyung Hee for her support and patience.

I. INTRODUCTION

The integrated services digital network (ISDN) is a planned worldwide public telecommunication network that will serve a wide variety of user needs. Error-free transmission is a necessity in any telecommunication network. The analysis and evaluation of errors that occur during the transmission have therefore received much attention in the digital communications industry.

Bit errors may be caused by distortion, thermal noise, timing jitter, hardware malfunction, software bugs, environment and man-made interferences, and other factors. These errors are present in the live traffic data bit stream. When the bit error rates exceed 10^{-6}, the bit error perturbations become noticeable in voice grade services and become unacceptable at 10^{-3} [Ref. 1][Ref. 9].

Many digital signal 1 (DS1) systems are characterized as having a combination of random errors and burst errors. Random error analysis forms the basis of most digital network maintenance and action limits. However, burst conditions, where high error rates occur for short durations, can impact services. It is important that such burst conditions be detected and corrected quickly to provide quality digital networks as digital services become an integral part of the user's operations.

The burst error is generated in digital networks by various unpredictable conditions. In addition, burst error patterns are always complicated by the pattern of the bit streams that are being transmitted. Although Dr. J. Eu has investigated the random error problem in [Ref. 1], the burst error has not been studied yet. Therefore, under the memoryless binary symmetric channel (MBSC) assumptions, the performance evaluation or estimation schemes of burst errors for DS1 transmission systems carrying live traffic is an interesting and important problem. The symmetric binary

channel is memoryless; a sequence of indepedent errors produces the noise digits. Each bit has the same probability of having an error occur.

To completely describe a burst error, one must know the bit pattern. This is impossible in practice on working systems, and consequently a burst error is characterized by a chosen set of independent parameters. Two commonly used parameters are bit error ratio (BER) and the length of the burst.

If one makes a vigorous examination of the number of burst errors and assumes that all error patterns are considered equally alike, the fraction of undetected burst errors tends to be upper-bounded by $1/2^b$ for a burst error length b [Ref. 1][Ref. 10]. The b is defined as any pattern of errors in a CRC block for which the number of bits between the first and last errors is inclusive.

In this study, Gilbert's model [Ref. 3] of a burst noise channel, which is a binary symmetric channel determined by a Markov chain, was used to derive the definition of burst error under the MBSC assumptions. A Markov chain with two states can be used to generate the burst errors. The two states will be called G (for Good) and B (for Bad or Burst). In state G the noise digits are always 0, meaning no noise. The Markov chain can be thought of as a Bernoulli trial of tossing a coin. In state B a coin is tossed to decide whether the noise digit will be 0 or 1. If the coin is fair the bit error ratio is 0.5. The exact probability depends on the bit error ratio. The coin-tossing feature is included because actual burst errors contain good digits interspersed with the errors. One may also use the mathematical model to derive the definition of burst error. An error burst can be defined as a string of bits with at least the first and the last bits in error for a bit stream of length b.

A detailed explanation of both methods to derive the definition of burst error will be discussed in Section II. In Section II Elliott's model [Ref. 6] is introduced for estimating the error rates for each code on burst noise channels. The efficient

detecting algorithms of the burst error is discussed for various cyclic redundancy check (CRC) codes, and show the undetected burst error patterns for each CRC code.

Analysis and evaluation of the estimation schemes based on the extended superframe (ESF) format using the probability density function (*pdf*) is discussed in Section III. The *pdf* of burst error of length b is proposed for each CRC block. The statistics of the mode, mean, variance, and standard deviation(SD) of each CRC code are also discussed in Section III.

In Section IV, the relationship between burst errors and block burst errors, and plots for each CRC code are discussed. Two different methods of simulating the burst error statistics, using binomial and exponential distribution, are investigated as well. Finally, concluding remarks are given in Section V.

II. DETECTING THE BURST ERROR

This chapter gives the definition of burst error, the algorithm of CRC error detection, and the pattern generated for the undetected burst errors.

A. DEFINITION OF BURST ERROR

Models have been used in the past to define burst error. The models most commonly used for this purpose are Gilbert's [Ref. 3] and Elliott's [Ref. 6]. An overview of both these models is presented, followed by the mathematical representation of a burst error.

Gilbert's model is based on a Markov chain. Gilbert used the burst noise model to define the burst error: the symmetric binary channel generates a sequence of binary noise digits. The burst noise channel consists of a two-state Markov chain which can be used to generate the burst error. The two states are G (for Good) and B (for Bad or Burst). In general, the channel state with the lower error probability is the good state. In order to complete the model one needs to specify the transition probabilities Q and q. Q is the conditional probability that the channel remains in the good state G for the next time interval, given that it is in state G at the present time interval. The q is the conditional probability that the channel remains in the bad state [Ref. 5].

To simulate the burst noise, the transition probabilities $P = Prob(G \rightarrow B)$ and $p = Prob(B \rightarrow G)$ will be small while Q and q are relatively larger. Runs of G will alternate with runs of B. The run lengths have geometric distributions with mean $1/P$ for the *G-runs* and mean $1/p$ for the *B-runs*. The geometric distribution of *G-runs* seems reasonable. If the various clicks, pops, and crashes, which might cause errors

on a real channel, are not related to one another, the times between such events will have a geometric distribution. Only mathematical simplicity justifies the geometric distribution of *B-runs*. The fraction of time spent in state B is $P(B) = P/(p + P)$. Since errors occur only in state B, and then just with probability $1 - h$, the error probability is

$$P(1) = (1 - h)P(B) = (1 - h)\frac{P}{p + P} \tag{2.1}$$

where h stands for making no error in state B.

As expected, long runs of good digits separate the burst errors. Therefore, we define a burst error as any contiguous sequence of transmission intervals during which the channel remains in the burst state. Basically, the burst error is merely a period of higher than average error probability.

Elliott's model is slightly different from Gilbert's model, in which an error bit can occur only when the channel is in the bad state. In the Elliott model, however, an error bit can occur in either the good or the bad state but with different probabilities. Transition between the good and bad states is the same as in the Gilbert model. Elliott's model uses the probabilities $P(m, n)$ to provide a means for estimating error rates for binary block codes in more general circumstances. $P(m, n)$ is the probability that m bit errors occur in a transmitted block of n bits. Elliott's model uses four different parameters (P, p, h, k) for computing the $P(m, n)$ where k is the probability of correct reception of a bit when the channel is in the good state and other parameters are the same as in the Gilbert model. Since an error bit can occur either in the good or the bad states, Elliott uses two different probabilities, $G(m, n)$ and $B(m, n)$, where $G(m, n)$ = Prob(m errors in a block of length n — the channel is in the good state at the first bit) and $B(m, n)$ = Prob(m errors in a block of length n — the channel

is in the bad state at the first bit). Then

$$P(m,n) = \frac{p}{P+p}G(m,n) + \frac{P}{P+p}B(m,n) \tag{2.2}$$

and $G(m,n)$ and $B(m,n)$ are given in [Ref. 6].

In a mathematical sense, a digital burst error can be defined as a string of bits with at least the first and the last bits in error over a length b [Ref. 4]. Traditionally one may express a bit stream by a polynomial [Ref. 7].

In the following discussion, one assumes that the burst error signal is $E(x)$. Then

$$E(x) = E(i,b,x) \tag{2.3}$$

where i determines how far from the right-hand end of the received frame the burst error is started and b is the burst error length. Such a polynomial is said to be of degree $b-1$. For example, a burst error of 100101 can be represented by a polynomial $E(x) = x^i(x^5 + x^2 + 1)$ with $b = 6$. From expression (2.3), it becomes

$$E(x) = x^i(1 + a_1x^1 + a_2x^2 + ... + a_{b-2}x^{b-2} + x^{b-1}) \tag{2.4}$$

where the first bit and the last bit are always 1 and other terms are either 0(Good) or 1(Error). Therefore the burst error signal can be rewritten as

$$E(x) = x^i(x^{b-1} + \sum_{j=1}^{b-2} a_jx^j + 1) \tag{2.5}$$

where $a_j \in [0,\ 1]$.

Using Equation 2.5 one can simulate the burst error behavior and evaluate its performance by examining the probability of burst error detection. For example, if the degree of $E(x)$ is less than the degree of CRC-n terms, the remainder can never be zero; hence the burst error is always detected. To be undetected, the burst error length b must be greater than n [Ref. 1][Ref. 7].

Among the three burst error models, the mathematical model is used in this thesis since in the evaluation of Chapter II and IV only the burst error length and CRC codes are the concerns. In this thesis, BER(ϵ) is used for generating the burst error and CRC codes for computing the probability of undetected burst error. That is,

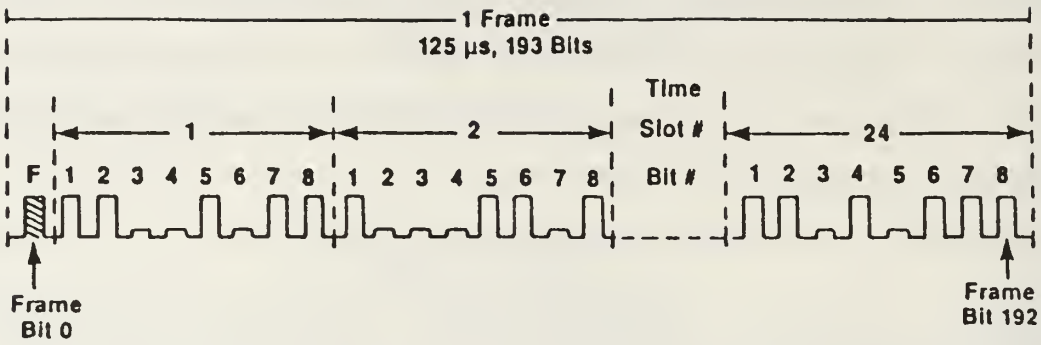
$$\epsilon = \frac{p}{p+P}k' + \frac{P}{p+P}h' \tag{2.6}$$

where $k' = 1 - k$ and $h' = 1 - h'$ in Elliott's model. Since only ϵ is used in this study, the values of P, p, h, and k are irrelevant, and the mathematical model is sufficient for the study in the rest of the thesis.

B. DETECTING ALGORITHM OF BURST ERROR

The CRC is a method of detecting the existence of errors in the transmission of digital signals. In communication systems using CRC codes, the digital signal to be transmitted is partitioned into CRC blocks of equal length. The bits of each block can be used to represent coefficients of terms in a polynomial, which are then divided by a standard divisor polynomial. The remainder of the division is then appended to the data block, and the resultant block is sent to the receiving end. The receiver performs the identical division on the data block as received and generates its own remainder. The receiver then compares its locally calculated remainder to the remainder transmitted with the block. A difference between the remainders indicates that the received block of the digital signal contained an error. Because each CRC code has its own generator polynomial, it is natural to think that the burst error pattern of each CRC code should be different. Before continuing the development of the detecting algorithm, the DS1 ESF format and CRC codes are discussed.

A DS1 signal has a nominal line rate of $1.544 Mbps$. A DS1 frame consists a set of 193 contiguous time slots (bits). The first bit in a DS1 frame is the framing bit, or



Note: Frame bit 0 is transmitted first.

Figure 2.1: Channelized DS1 Frame Bit Assignments

"F-bit." The remaining 192 bits of a DS1 frame are labeled "information" bits and are collectively referred to as the DS1 "payload." These 192 bits may form a single $1536 Kbps$ channel, or they may be divided into a variety of partitions, including the common 24 8-bit words corresponding to 24 separate voice grade channels. Collectively these are often referred to as "channelized" applications. Figure 2.1 illustrates this configuration of a DS1 frame [Ref. 2].

One ESF format is composed of 24 consecutive DS1 frames. Each frame is an ordered sequence of 193 bits. Therefore, the 24 DS1 frames represent 4632 bits, and the time duration of the ESF is $3ms$. The first bit of each frame (framing bit) is designated for frame synchronization, data communication, and CRC-6 error detection. The ESF format is shown in TABLE 2.1 [Ref. 2]. The sequence of F-bits in contiguous frames forms an *8-kb/s* channel, and the fixed frame pattern sequence (FPS) is 001011, which appears in the first bits of frames 4, 8, 12, 16, 20, and 24 of the superframe. Each ESF has the six CRC bit positions which appear in frames 2, 6, 10, 14, 18, and 22.

TABLE 2.1: EXTENDED SUPERFRAME FORMAT

Frame No.	F Bit			
	Bit No.	FPS	CRC	DL
1	0	-	-	X
2	193	-	C1	-
3	386	-	-	X
4	579	0	-	-
5	772	-	-	X
6	965	-	C2	-
7	1158	-	-	X
8	1351	0	-	-
9	1544	-	-	X
10	1737	-	C3	-
11	1930	-	-	X
12	2123	1	-	-
13	2316	-	-	X
14	2509	-	C4	-
15	2702	-	-	X
16	2895	0	-	-
17	3088	-	-	X
18	3281	-	C5	-
19	3474	-	-	X
20	3667	1	-	-
21	3860	-	-	X
22	4053	-	C6	-
23	4246	-	-	X
24	4439	1	-	-

In ESF, the block of data over which a CRC is calculated is one ESF. The standard divisor used is the sixth order polynomial: $x^6 + x + 1$. This divisor results in a 6-bit remainder sometimes referred to as the "CRC-6 code." It is this remainder that is carried in the six check bits, C1 through C6, of the CRC channel. The CRC-6 code calculated for a particular ESF is always carried in the CRC channel of the subsequent ESF [Ref. 2].

The following is a detailed description of the steps required to compute the

CRC-6 code [Ref. 2][Ref. 7]:

1. For the purpose of calculating the CRC-6 code of an extended superframe, the 24 F-bits of that ESF shall temporarily be set to the binary value "1." For transmission, the true value of the F-bits is restored.

2. The resulting 4632 bits of the ESF are used in order of occurence to construct a polynomial in x such that bit number 0 is the coefficient of the term x^{4631}, and bit number 4631 is the coefficient of the term x^0.

3. This polynomial is multiplied by the factor x^6, and the result is divided by the generator polynomial $x^6 + x + 1$. The binary coefficients of the remainder polynomial are used in order of occurence as bits C1 through C6 of the CRC channel in the subsequent ESF. The ordering is such that the coefficient of the term x^5 in the remainder polynomial is bit C1, and the coefficient of the term x^0 in the remainder polynomial is bit C6.

4. The bits C1 through C6 contained in an ESF shall always be those associated with the content of the ESF immediately preceeding ESF. Bits C1-C6 may be assigned any value.

The CRC method of DS1 described here is capable of detecting all the ESFs with six or fewer errors. The probability of all ESFs containing more than six transmission errors is 63/64(98.4%). The CRC method does not indicate the number of bit errors in an ESF, but only indicates that there was at least one. The current usage of CRC codes for error performance monitoring are illustrated in TABLE 2.2 [Ref. 7][Ref. 10]. This table contains the block size in bits and the generator polynomial for each CRC code. The CRC-12 system is used for transmission of streams of 6-bit characters and generates a 12-bit frame check sequence (FCS). Both CRC-16 ANSI and CRC-16 CCITT are popular for 8-bit characters, in the United States and Europe respectively, and both result in a 16-bit FCS [Ref. 8].

TABLE 2.2: CRC CODES

CRC codes	Block size	Generator polynomial
CEPT1 CRC-4	2048	$x^4 + x + 1$
DS1 CRC-6	4614	$x^6 + x + 1$
DS3 CRC-9	14280	$x^9 + x^4 + 1$
CRC-12	1880	$x^{12} + x^{11} + x^3 + x^2 + x + 1$
CRC-16 ANSI	256, 1024	$x^{16} + x^{15} + x^2 + 1$
CRC-16 CCITT	256, 1024	$x^{16} + x^{12} + x^5 + 1$

It is interesting to derive the sequence of the undetected burst error pattern and why the probability of an undetected burst error is always $1/2^{b-2}$. Using the generator polynomial $G(x)$ (see TABLE 2.2) and Equation 2.5, for an undetected burst error, $E(x) = 0 \bmod G(x)$, i.e., $E(x)$ will be divisible by $G(X)$. Suppose $E(x)/G(x) = Q(x)$ for the undetected burst error polynomial $E(x)$. Then, one can derive $E(x)$ by $G(x)Q(x)$ for various $Q(x)$. If $E(x)$ is of degree j (the burst error length $j + 1$) and the $G(x)$ degree is r for a given CRC-r code, then $Q(x)$ should have a degree of $j - r$ when $E(x)$ is undetected. Thus, for any undetected burst error length, one can construct $E(x)$ from $Q(x)G(x)$ for given all possible $Q(x)$ with given burst error lengths.

Example: For $Q(x)$ of degree 3, there are 4 possible patterns, 1001, 1011, 1101, and 1111. All $E(x)$ of degree 9 (burst error length 10) therefore are generated, which means that there are 4 undetected burst errors with length 10 for the CRC-6 code. The degree of $E(x)$ is (*the degree of* $Q(x)$) + (*the degree of* $G(x)$). For the ratio of the portion of each burst error length n, the undetected burst error is always 1/64. However, $n = 7$ is a special case due to CRC-6. In other words, the probability of detected burst error for any number of burst error lengths and each CRC code is always 63/64(98.4%).

The bit patterns of the undetected burst errors for the CRC-6 code are shown in TABLE 2.3. In TABLE 2.3, the leftmost bit is the most significant bit (MSB) and the rightmost bit is the least significant bit (LSB). If one examines TABLE 2.3, the number of undetected burst errors is 4 for the burst error length 10. Since we have 2^{10-2} possible $E(x)$ where the number 2 is the first and last bit position and always 1, the probability of undetected burst error is $4/2^8 = 1/64 = 0.015625$. The probability of detected burst error is $1 - 0.015625 = 0.9844(98.44\%)$. Therefore the probability of detected burst error for any burst error length is always $63/64(98.4\%)$. The bit pattern of undetected burst errors for other burst error lengths and various CRC codes are shown in APPENDIX A. It can be proven that the detection probability for any burst error length and each CRC code is always $1 - 1/2^n$ where n is the CRC degree.

In TABLE 2.2, the generator polynomials of CRC-12, CRC-16 CCITT, and CRC-16 ANSI contain $(x + 1)$ as a factor, and this ensures that all odd-weight burst errors will be detected [Ref. 10][Ref. 11]. By computing the weight of the undetected burst error patterns which are shown in APPENDIX A, one knows that all the burst errors which have odd-weight in these three CRC codes are detected.

TABLE 2.3: UNDETECTED BURST ERROR PATTERN FOR CRC-6 CODE

Burst Error Length	Undetected Error Pattern
7	1 0 0 0 0 1 1
8	1 1 0 0 0 1 0 1
9	1 0 1 0 0 1 1 1 1 1 1 1 0 0 1 0 0 1
10	1 1 0 1 0 1 0 1 1 1 1 0 0 1 0 1 1 0 1 1 1 0 1 1 0 1 1 1 0 1 1 1 1 1 0 1 0 0 0 1
11	1 0 1 0 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 1 1 1 1 1 0 0 1 1 0 1 0 1 1 1 0 0 0 1 1 1 0 0 1 1 1 1 0 1 1 1 0 1 1 0 1 1 0 0 1 1 1 1 0 1 0 1 1 0 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 0 0 0 0 1

III. EVALUATING ESTIMATION SCHEMES

Assume the MBSC whose crossover error probability (BER) is ϵ. Then, for each CRC block of n bits, the probability density function $f(b)$ of burst error length b can be stated as follows:

$$f(b) = \begin{cases} \frac{(n+1-b)\epsilon^2(1-\epsilon)^{n-b}}{1-(1-\epsilon)^n} & \text{if } 2 \leq b \leq n \\ \frac{n\epsilon(1-\epsilon)^{n-1}}{1-(1-\epsilon)^n} & \text{if } b = 1 \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

Notice that $(1 - \epsilon)^n$ is the probability of zero errors. Hence $1 - (1 - \epsilon)^n$ is the probability that an error occurs, or the total error probability.

One now has proof that the *pdf* of the above equation is correct. Consider a block of n bits, each of which has an error probability ϵ. The total probability is derived for each of the two cases: $b = 1$ and $b > 1$. Let $g(b)$ be the total probability that a burst error of length b occurs in the block.

For $b = 1$, the burst error is a single error by definition and thus the total probability for this case is given by

$$g(b) = n\epsilon(1 - \epsilon)^{n-1} \tag{3.2}$$

For $2 \leq b \leq n$, let $b = k$, then the burst error of length $b = k$ is of the form $ex \cdots xe$ where e denotes an error and x denotes any single bit whether it is in error or valid. The number of all distinct bit positions that a burst error of length k may start within a block of n bits is (n - k + 1). Thus, the total probability for $b = k$ is given by

$$g(k) = (n+1-k)\epsilon^2(1-\epsilon)^{n-k} \tag{3.3}$$

To complete the proof, we need to show that the sum of all the total probabilities is $1-(1-\epsilon)^n$. Noting that

$$\sum_{i=1}^{n-1} i(1-\epsilon)^{i-1} = \epsilon^{-2}[1-n(1-\epsilon)^{n-1}+(n-1)(1-\epsilon)^n] \tag{3.4}$$

and rearranging the indices, we now have

$$\sum_{k=2}^{n} g(k) = \epsilon^2 \sum_{i=1}^{n-1} i(1-\epsilon)^{i-1} = 1-n(1-\epsilon)^{n-1}+(n-1)(1-\epsilon)^n. \tag{3.5}$$

Since

$$\sum_{k=1}^{n} g(k) = 1-(1-\epsilon)^n, \tag{3.6}$$

we get

$$\sum_{k=1}^{n} f(k) = 1. \tag{3.7}$$

This completes the proof.

From Equation 3.1, one can compute the mean, SD, F(6) and two types of modes for CRC-6 code which are summarized in TABLE 3.1. In TABLE 3.1, the F(6) depends only on the BER and is given by

$$F(6) = \sum_{k=1}^{6} f(k), \tag{3.8}$$

TABLE 3.1: MEANS, SD, F(6), AND MODE FOR CRC-6 CODE

BER	Mean	SD	F(6)	Mode($b \geq 7$)	Mode($b \geq 1$)
10^{-8}	1.036	9.047	1.0000	7	1
10^{-7}	1.355	28.61	0.9998	7	1
10^{-6}	4.548	90.43	0.9977	7	1
10^{-5}	36.48	284.6	0.9772	7	1
10^{-4}	354.6	850.3	0.7874	7	1
10^{-3}	2708	1202	0.0464	3615	3615
10^{-2}	4416	140.7	0.0000	4515	4515
10^{-1}	4596	13.42	0.0000		

where the value 6 is the degree of $G(x)$ in CRC-6, and the mode means that the value of k is $P(b = k)$ maximized. The distribution of burst error between the burst error lengths, and $f(b)$ is depicted in Figure 3.1 with $b \geq 1$ and in Figure 3.2 with $b \geq 7$. The mean of the burst error length for BER = 10^{-8} is much smaller than for BER = 10^{-1}. This means that the detected burst error length depends only on the BER. When the BER is less than 10^{-6} for CRC-6 codes, the mean burst error lengths are 4.548, 1.355, and 1.036. They are most likely detected by the generator polynomial $G(x)$ because their lengths are less than 6. By the definition of mode, the burst error lengths 1 and 7 occur more frequently for a BER less than 10^{-4} for $b \geq 1$ and $b \geq 7$. It is very interesting to note that as seen in TABLE 3.1, in the vicinity of BER 10^{-3} the SD of the burst error length seems to achieve the maximum value. One suspects that the SD will be at maximum in the vicinity of 1/4614 for the CRC-6 code since 1/4614 is close to 10^{-3}. Similar observations are obtained for other CRC blocks.

It seems worthwhile to investigate the relationship between the mean burst error length and CRC block size with BER. The computation result is summarized in TABLE 3.2, and the corresponding plot is depicted in Figure 3.3. The X-axis stands

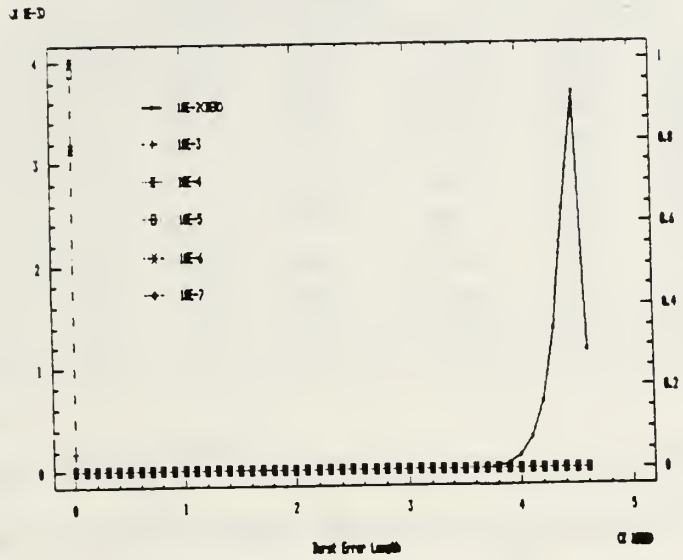


Figure 3.1: The Distribution of Burst Error with $b \geq 1$ for CRC-6 Code

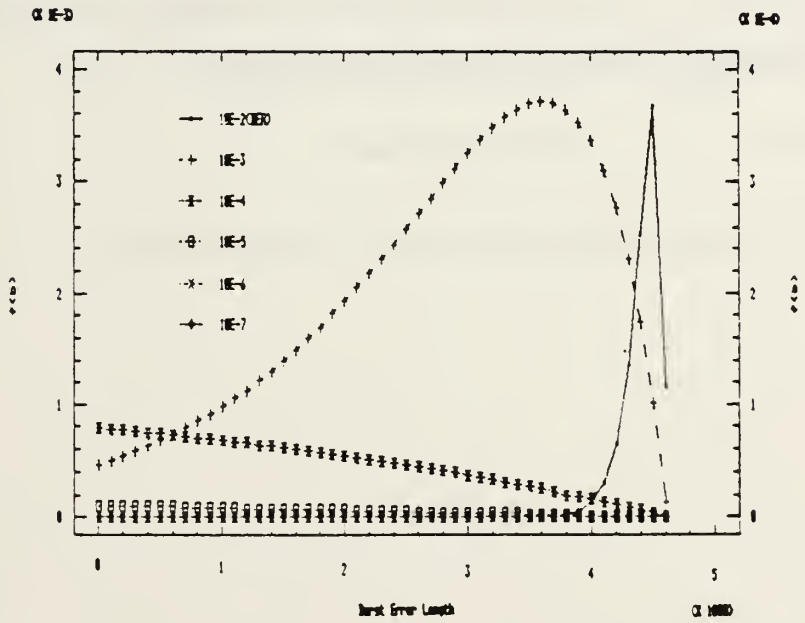


Figure 3.2: The Distribution of Burst Error with $b \geq 7$ for CRC-6 Code

TABLE 3.2: MEAN BURST ERROR LENGTH WITH BER

BER	CRC Block Size							
		CRC16	CRC16	CRC16	CRC4	CRC6		CRC9
	128	256	512	1024	2048	4614	8192	14280
10^{-8}	1.000	1.000	1.000	1.000	1.007	1.036	1.112	1.340
10^{-7}	1.000	1.001	1.004	1.017	1.070	1.355	2.118	4.399
10^{-6}	1.003	1.011	1.044	1.175	1.699	4.548	12.18	34.99
10^{-5}	1.027	1.109	1.437	2.748	7.990	36.48	112.8	340.8
10^{-4}	1.273	2.092	5.369	18.47	70.86	354.6	1107	3290
10^{-3}	3.731	11.92	44.52	172.9	655.9	2708	6199	12282
10^{-2}	27.71	100.3	320.0	826.1	1850	4416	7994	14082

for the CRC block size in bits (128 through 15,000) and the Y-axis stands for the mean burst error length in bits (0 through 15,000). It is clear that the mean burst error length is dependent only on the CRC block size, not on the bit position where the burst error occurs. When BER is less than 10^{-6}, the mean burst error length is almost independent of CRC block size as compared with the higher BER (i.e., BER is greater than 10^{-5}). The detection probability is completely dependent on the CRC block size, regardless of the CRC generator polynomial.

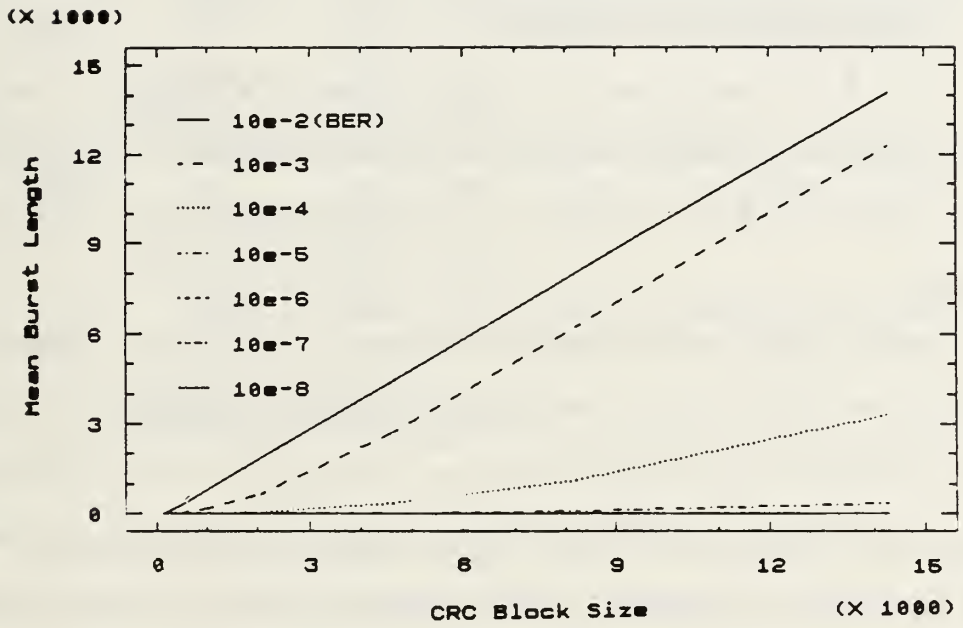


Figure 3.3: The Distribution of Mean Burst Error Length with BER

IV. ANALYSING ESTIMATION SCHEMES

This chapter gives the simulation results of the burst error for each CRC block and CRC code using binomial and exponential distribution for generating the random number.

A. BINOMIAL DISTRIBUTION

Let X be a binomial random variable. Then X is the sum of the Bernoulli random variable associated with each of the n independent trials, where n is the length of the burst error. The error occurrences in different bits are independent of each other.

Using this fact, one can simulate the burst error for different CRC blocks and different types of CRC codes. First, the random number is generated, bit by bit, then the bit is checked to see whether an error has occurred or not. Second, the burst error length is divided into each CRC block and we generate the block burst error length using the definition of burst error. The block burst error is a burst error contained in specified block such as the CRC-6 block (4614 bits). Lastly, the statistical results are computed for the block burst error length. This process continues for a sufficient number of randomly generated burst errors.

The results of the block burst error for CRC-6 using the binomial distribution are shown in TABLE 4.1. The relationship between the total burst error length and the block burst errors with various BERs for CRC-6 is shown in TABLE 4.1. The SD of the block burst error length has a maximum value in the vicinity of BER 10^{-3} for the different total burst error lengths. This means that the SD will be at a maximum in the vicinity of 1/4614 for CRC-6. This phenomenon occurs with other CRC codes

TABLE 4.1: SIMULATION RESULTS OF CRC-6 USING BINOMIAL DISTRIBUTION

Total Burst Length	BER / Statistical	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
15000	Mean	3448	1250	14.12	1.149	1.001	1.000
	SD	140.7	859.7	64.45	2.830	0.042	0.000
	Mode($b > 0$)	3409	1	1	1	1	1
	Mode($b > 6$)	3409	686	7	7	7	0
20000	Mean	3693	1410	18.16	1.048	1.001	1.000
	SD	140.7	914.6	79.37	1.106	0.063	0.030
	Mode($b > 0$)	3625	1	1	1	1	1
	Mode($b > 6$)	3625	1027	7	7	7	7
25000	Mean	3863	1534	14.70	1.080	1.004	1.000
	SD	140.7	954.7	68.30	1.868	0.249	0.000
	Mode($b > 0$)	3825	1	1	1	1	1
	Mode($b > 6$)	3825	947	7	7	7	7
300000	Mean	4250	1779	15.42	1.024	1.000	1.000
	SD	140.7	1027	79.71	1.080	0.021	0.000
	Mode($b > 0$)	4360	1	1	1	1	1
	Mode($b > 6$)	4360	2298	7	7	7	7

as well. The higher BER has a longer length of mean block burst error than the lower BER. This indicates that the probability of detected block burst error for a lower BER is higher than the higher BER which is consistent with intuitive reasoning.

It is also intersesting to note that when the BER gets larger, then the value of the mode is larger for both cases, i.e., $b > 0$ and $b > 6$. This means that one cannot predict which block burst error lengths are expected for each CRC block. But when the BER gets lower, the value of the mode is small. Thus one can guess that the block burst error is a single bit and can expect that the probability of detected block burst error is 63/64(98.4%). In other words, a mode of large value such as 3409 makes the prediction of burst error length more difficult since the uncertainity is higher than the case of a mode value of 1.

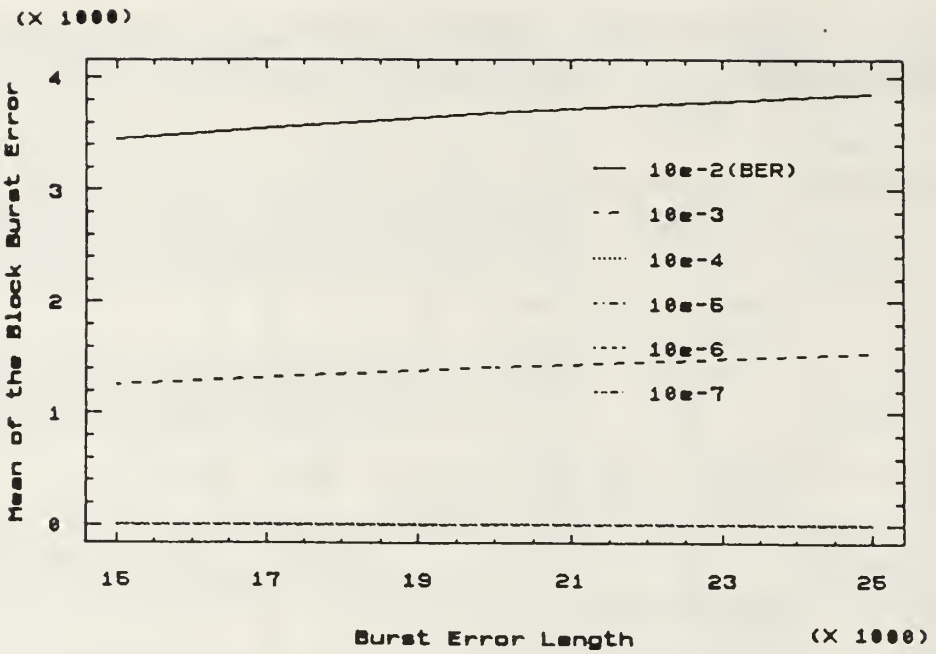


Figure 4.1: Distribution of CRC-6 using Binomial Distribution

The relationship between the burst errors and block burst errors for each CRC code is depicted in Figure 4.1. The X-axis stands for the total burst error length in bits and the Y-axis stands for the mean of the block burst errors in bits. The mean block burst error depends on the BER and has a very small value as the BER gets lower, as depicted in Figure 4.1. The simulation results for the different CRC codes that are listed in TABLE 2.2 using binomial distribution are shown in APPENDIX B.

B. EXPONENTIAL DISTRIBUTION

The relationship between the binomial distribution and exponential distribution will now be explained for this study. The Poisson random variable arises in situations where the burst errors occur "completely at random" in one ESF for each type of CRC block. The *pdf* for the Poisson random variable is given by

$$P[N = k] = \frac{\alpha^k}{k!}\epsilon^{-\alpha} \qquad k = 0, 1, 2, \cdots \tag{4.1}$$

where α is the average number of the burst error occurence in an ESF for each type of CRC block.

One of the applications of the Poisson probabilities in Equation 4.1 is to approximate the binomial probabilities. If n is large and p is small, then for $\alpha = np$

$$P_k = \binom{n}{k} p^k (1-p)^{n-k} \simeq \frac{\alpha^k}{k!} e^{-\alpha} \qquad \text{for } k = 0, 1, 2, \cdots \tag{4.2}$$

where n is the total burst error length and p is the BER. Thus the Poisson *pdf* is the limiting form of the binomial *pdf* when the number of Bernoulli trials n is made very large and the probability of success is kept small, so that $\alpha = np$ [Ref. 12].

For one ESF (4614 bits) for the CRC-6 code, the interval T bits has been divided into n bits. Each bit can be viewed as a Bernoulli trial if the following conditions hold: (1) At most one error can occur in a bit, (2) the outcomes in different subintervals are independent, and (3) the probability of an error occurrence in a bit is given by $p = \alpha/n$, where α is the average number of errors observed in a T bit interval. As $n \to \infty$, the occurrence of the first error approaches an exponential random variable with parameter $\lambda = \alpha/T$.

Therefore, the exponential random variable is obtained as a limiting form of the geometric random variable. This means that, for a Poisson random variable, the time between errors is an exponentially distributed random variable with parameter $\lambda = \alpha/T$. The exponential random variable satisfies the memoryless property. One can implement the exponential random variable using the inverse transformation method for the burst error [Ref. 13]. This method is generally used for distributions whose

cumulative distribution function (*cdf*) can be obtained in closed form.

The *pdf* of the exponential distribution with parameter λ is given by

$$f(x) = \begin{cases} \lambda e^{-\lambda x} & \text{if } x \geq 0, \lambda > 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.3}$$

and the *cdf* of $f(x)$ is

$$F(x) = \begin{cases} 0 & \text{if } x < 0 \\ 1 - e^{-\lambda x} & \text{if } x \geq 0 \end{cases} \tag{4.4}$$

Next, a random number r is generated and $F(x) = r$ is set for solving the x. This gives

$$1 - e^{-\lambda x} = r. \tag{4.5}$$

Rearranging to

$$e^{-\lambda x} = 1 - r \tag{4.6}$$

and taking the natural logarithm of both sides, we have

$$-\lambda x = \ln(1 - r). \tag{4.7}$$

Finally, solving for x gives the solution

$$x = -\frac{1}{\lambda} \ln(1 - r). \tag{4.8}$$

To simplify the computations, we can replace $(1-r)$ by r in Equation 4.8. Since r is a random number, $(1-r)$ will also be a random number. This means that nothing has changed except the way we are writing the $U(0,1)$ random number. Thus the process generator for the exponential distribution will now be

$$x = -\frac{1}{\lambda} \ln r. \tag{4.9}$$

From Equation 4.9, $\lambda = \frac{\alpha}{T}$, and $\alpha = np$, then

$$x = -\frac{1}{p} \ln r \tag{4.10}$$

TABLE 4.2: SIMULATION RESULTS OF CRC-6 USING EXPONENTIAL DISTRIBUTION

Total Burst Length	BER / Statistical	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
15000	Mean	3446	1198	15.71	1.184	1.000	1.000
	SD	140.7	840.0	69.32	3.392	0.000	0.000
	Mode($b > 0$)	3320	1	1	1	1	1
	Mode($b > 6$)	3320	494	7	7	0	0
20000	Mean	3705	1334	16.77	1.067	1.000	1.000
	SD	140.7	892.1	72.95	1.618	0.000	0.000
	Mode($b > 0$)	3667	1	1	1	1	1
	Mode($b > 6$)	3667	1069	7	7	0	0
25000	Mean	3870	1492	18.21	1.083	1.000	1.000
	SD	140.7	941.5	80.35	1.782	0.004	0.000
	Mode($b > 0$)	3949	1	1	1	1	1
	Mode($b > 6$)	3949	1150	7	7	7	0
300000	Mean	4246	1760	15.38	1.029	1.000	1.000
	SD	140.7	1022	79.63	1.235	0.023	0.000
	Mode($b > 0$)	4334	1	1	1	1	1
	Mode($b > 6$)	4334	2190	7	7	7	7

where p is the BER.

From Equation 4.10, one can simulate the burst error for the exponential distribution. The method of simulation is much like the binomial case except for the burst error generation. The results of the block burst error for CRC-6 using exponential distributions are shown in TABLE 4.2 and are depicted in Figure 4.2. It is very interesting to note that all the simulation results of the exponential distribution are similar to those of the binomial distribution. The simulation results for the different CRC codes that are listed in TABLE 2.2 using exponential distribution are shown in APPENDIX C.

Recall that TABLE 2.1 shows various CRCs and their block lengths for a fixed length CRC. The detectability (r) should be the ratio of block length and the length

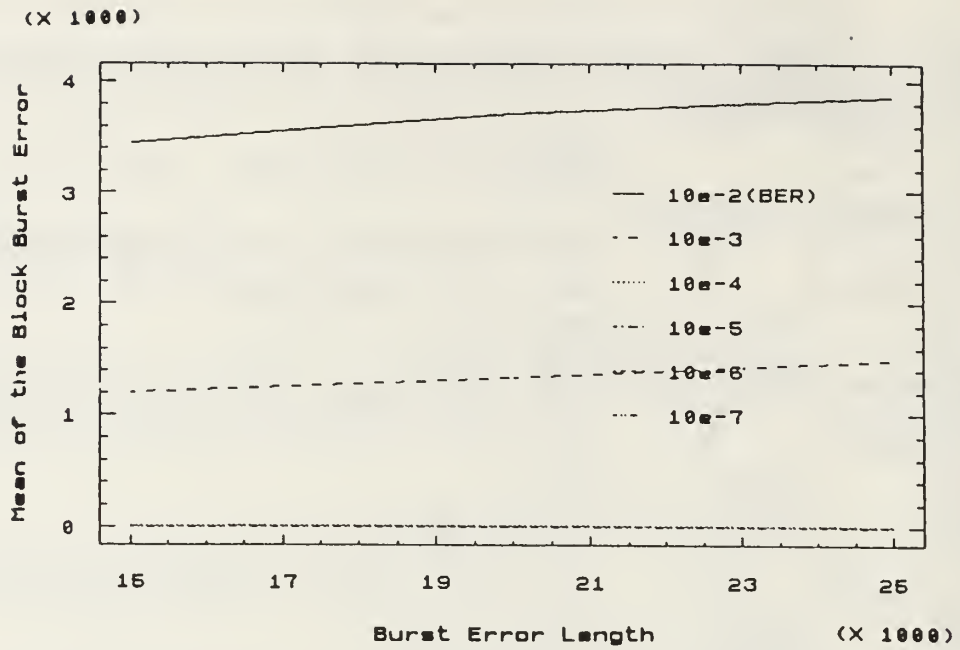


Figure 4.2: Distribution of CRC-6 using Exponential Distribution

of checkword. Note that the CRC-9 block length is too large compared with other CRCs: each bit in CRC-9 checkword must check 1587.67 bits, that is more than ten times higher than the CRC-12 scheme. This can be seen from the simulation result that even for a lower bit error rate, the error burst length of CRC-9 is longer than the CRC-12. For example, the mean burst error length of CRC-9 with BER 10^{-5} is 6.488, but the mean burst error length of CRC-12 with BER 10^{-4} is 1.706.

It should be noted that the SD of the block burst error length has a maximum value in the vicinity of BER 10^{-3} for the different burst error lengths and CRC codes except the CRC-16 (APPENDIX C). This means that the SD will be at a maximum in the vicinity of *1/(CRC block size)* for any CRC codes with the exception of CRC-16. In fact, this observation can apply to CRC-16, since 1/256 is in the BER 10^{-2} that is shown in APPENDIX C.

If one examines all the simulation results of both the binomial and exponential

cases, the mean block burst error depends only on the BER for a given CRC code. When the BER is less than 10^{-3}, the value of the mean burst length is very small as compared with the higher BER for any CRC codes. For this reason, the mean block burst error is independent of the CRC generator polynomial. Hence, when the burst error length gets longer and BER gets lower, the mean block burst error tends to approach the pattern of the burst error which the random bit errors generate. Because the burst error occurs randomly or is positionally independent, the detection probability is completely dependent on the CRC block size, regardless of the CRC generator polynomial.

V. CONCLUSIONS

As the undetected burst error bit patterns are generated, it can be proven that the detection probability for any burst error length and each CRC code is always $1-1/2^n$. The mean, SD, F(6), and two types of modes for CRC-n codes are computed using the probability density function.

The relationship between the mean burst error length and CRC block size with BER was investigated, and it was found that the mean burst error length is dependent only on the CRC block size, not on the bit position where the burst error occurs. The detection probability is completely dependent on the CRC block size, regardless of the CRC generator polynomial.

The burst errors for different CRC blocks and different types of CRC codes are simulated through the use of computer simulation within the CRC block size. The CRC block burst error for CRC-n using binomial and exponential distribution was evaluated. The simulation results show that the SD of the block burst error length has a maximum value in the vicinity of BER 10^{-3} for the different total burst error lengths and CRC codes except CRC-16. This means that the SD will be at a maximum in the vicinity of *1/(CRC block size)* for any CRC codes.

Finally, the mean block burst error is independent of the CRC generator polynomial and depends only on the BER for a given CRC code. Hence, when the burst error length gets longer and the BER gets lower, the mean block burst error tends to approach the pattern of the burst error which the random bit errors generate. Because the burst error occurs randomly or is positionlly independent, the detection probability is completely dependent on the CRC block size, regardless of the CRC generator polynomial.

APPENDIX A: UNDETECTED BURST ERROR BIT PATTERNS

The following lists are patterns of the undetected burst error. The leftmost bit is the most significant bit (MSB) and the rightmost bit is the least significant bit (LSB) for each CRC code. In each case the undetected burst error bit pattern is generated by computer and the burst error length is greater than the degree of generator polynomial $G(x)$. When the burst error length is smaller than the degree of $G(x)$, any burst error can be detected by $G(x)$. The source code for the undetected burst error bit pattern generator is shown in APPENDIX D. 1.

1. CRC-4(CEPT)

$G(x) = x^4 + x + 1$

```
burst lenth 5  : 1 0 0 1 1

burst lenth 6  : 1 1 0 1 0 1

burst lenth 7  : 1 0 1 1 1 1 1
                 1 1 1 1 0 0 1

burst lenth 8  : 1 1 0 0 0 1 1 1
                 1 0 0 0 1 0 1 1
                 1 0 1 0 1 1 0 1
                 1 1 1 0 0 0 0 1

burst lenth 9  : 1 0 1 1 0 1 1 1 1
                 1 1 1 1 1 0 1 1 1
                 1 1 0 1 1 1 0 1 1
```

```
                1 0 0 1 0 0 0 1 1
                1 1 0 0 1 1 1 0 1
                1 0 0 0 0 0 1 0 1
                1 0 1 0 0 1 0 0 1
                1 1 1 0 1 0 0 0 1
burst lenth 10 : 1 0 0 0 1 1 1 1 1 1
                1 1 0 0 0 0 1 1 1 1
                1 1 1 0 0 1 0 1 1 1
                1 0 1 0 1 0 0 1 1 1
                1 1 1 1 0 1 1 0 1 1
                1 0 1 1 1 0 1 0 1 1
                1 0 0 1 1 1 0 0 1 1
                1 1 0 1 0 0 0 0 1 1
                1 1 1 1 1 1 1 1 0 1
                1 0 1 1 0 0 1 1 0 1
                1 0 0 1 0 1 0 1 0 1
                1 1 0 1 1 0 0 1 0 1
                1 0 0 0 0 1 1 0 0 1
                1 1 0 0 1 0 1 0 0 1
                1 1 1 0 1 1 0 0 0 1
                1 0 1 0 0 0 0 0 0 1
```

2. CRC-6

$$G(x) = x^6 + x + 1$$

```
burst lenth 7  : 1 0 0 0 0 1 1
```

```
burst lenth 8  : 1 1 0 0 0 1 0 1

burst lenth 9  : 1 0 1 0 0 1 1 1 1

                 1 1 1 0 0 1 0 0 1

burst lenth 10 : 1 1 0 1 0 1 0 1 1 1

                 1 0 0 1 0 1 1 0 1 1

                 1 0 1 1 0 1 1 1 0 1

                 1 1 1 1 0 1 0 0 0 1

burst lenth 11 : 1 0 1 0 1 1 1 1 1 1 1

                 1 1 1 0 1 1 0 0 1 1 1

                 1 1 0 0 1 1 0 1 0 1 1

                 1 0 0 0 1 1 1 0 0 1 1

                 1 1 0 1 1 1 0 1 1 0 1

                 1 0 0 1 1 1 1 0 1 0 1

                 1 0 1 1 1 1 1 1 0 0 1

                 1 1 1 1 1 1 0 0 0 0 1

burst lenth 12 : 1 1 0 1 0 0 0 1 1 1 1 1

                 1 0 0 1 0 0 1 0 1 1 1 1

                 1 0 1 1 0 0 1 1 0 1 1 1

                 1 1 1 1 0 0 0 0 0 1 1 1

                 1 0 1 0 0 0 1 1 1 0 1 1

                 1 1 1 0 0 0 0 0 1 0 1 1

                 1 1 0 0 0 0 0 1 0 0 1 1

                 1 0 0 0 0 0 1 0 0 0 1 1

                 1 0 1 0 1 0 1 1 1 1 0 1

                 1 1 1 0 1 0 0 0 1 1 0 1

                 1 1 0 0 1 0 0 1 0 1 0 1
```

1 0 0 0 1 0 1 0 0 1 0 1

1 1 0 1 1 0 0 1 1 0 0 1

1 0 0 1 1 0 1 0 1 0 0 1

1 0 1 1 1 0 1 1 0 0 0 1

1 1 1 1 1 0 0 0 0 0 0 1

3. CRC-9(DS3)

$G(x) = x^9 + x^4 + 1$

burst lenth 10 : 1 0 0 0 0 1 0 0 0 1

burst lenth 11 : 1 1 0 0 0 1 1 0 0 1 1

burst lenth 12 : 1 1 1 0 0 1 1 1 0 1 1 1

1 0 1 0 0 1 0 1 0 1 0 1

burst lenth 13 : 1 1 1 1 0 1 1 1 1 1 1 1 1

1 0 1 1 0 1 0 1 1 1 0 1 1

1 1 0 1 0 1 1 0 1 1 1 0 1

1 0 0 1 0 1 0 0 1 1 0 0 1

burst lenth 14 : 1 1 1 1 1 1 1 1 1 0 1 1 1 1

1 0 1 1 1 1 0 1 1 0 0 1 1 1

1 1 0 1 1 1 1 0 1 0 1 0 1 1

1 0 0 1 1 1 0 0 1 0 0 0 1 1

1 1 1 0 1 1 1 1 0 0 1 1 0 1

1 0 1 0 1 1 0 1 0 0 0 1 0 1

1 1 0 0 1 1 1 0 0 0 1 0 0 1

1 0 0 0 1 1 0 0 0 0 0 0 0 1

burst lenth 15 : 1 0 1 1 1 0 0 1 1 0 1 1 1 1 1

1 1 1 1 1 0 1 1 1 0 0 1 1 1 1

1 0 0 1 1 0 0 0 1 0 1 0 1 1 1

1 1 0 1 1 0 1 0 1 0 0 0 1 1 1

1 0 1 0 1 0 0 1 0 0 1 1 0 1 1

1 1 1 0 1 0 1 1 0 0 0 1 0 1 1

1 0 0 0 1 0 0 0 0 0 1 0 0 1 1

1 1 0 0 1 0 1 0 0 0 0 0 0 1 1

1 0 1 1 0 0 0 1 1 1 1 1 1 0 1

1 1 1 1 0 0 1 1 1 1 0 1 1 0 1

1 0 0 1 0 0 0 0 1 1 1 0 1 0 1

1 1 0 1 0 0 1 0 1 1 0 0 1 0 1

1 0 1 0 0 0 0 1 0 1 1 1 0 0 1

1 1 1 0 0 0 1 1 0 1 0 1 0 0 1

1 0 0 0 0 0 0 0 0 1 1 0 0 0 1

1 1 0 0 0 0 1 0 0 1 0 0 0 0 1

4. CRC-12

$$G(x) = x^{12} + x^{11} + x^3 + x^2 + x + 1$$

burst lenth 13 : 1 1 0 0 0 0 0 0 0 1 1 1 1

burst lenth 14 : 1 0 1 0 0 0 0 0 0 1 0 0 0 1

burst lenth 15 : 1 1 1 1 0 0 0 0 0 1 1 0 0 1 1

1 0 0 1 0 0 0 0 0 1 0 1 1 0 1

burst lenth 16 : 1 1 0 1 1 0 0 0 0 1 1 1 0 1 1 1

1 0 1 1 1 0 0 0 0 1 0 0 1 0 1 1

1 0 0 0 1 0 0 0 0 1 0 1 0 1 0 1

```
                1 1 1 0 1 0 0 0 0 1 1 0 1 0 0 1
burst lenth 17 : 1 1 0 0 1 1 0 0 0 1 1 1 1 1 1 1 1
                1 0 1 0 1 1 0 0 0 1 0 0 0 0 1 1 1
                1 0 0 1 1 1 0 0 0 1 0 1 1 1 0 1 1
                1 1 1 1 1 1 0 0 0 1 1 0 0 0 0 1 1
                1 1 1 0 0 1 0 0 0 1 1 0 1 1 1 0 1
                1 0 0 0 0 1 0 0 0 1 0 1 0 0 1 0 1
                1 0 1 1 0 1 0 0 0 1 0 0 1 1 0 0 1
                1 1 0 1 0 1 0 0 0 1 1 1 0 0 0 0 1
burst lenth 18 : 1 0 1 0 0 1 1 0 0 1 0 0 0 1 1 1 1 1
                1 1 0 0 0 1 1 0 0 1 1 1 1 0 1 1 1 1
                1 1 1 1 0 1 1 0 0 1 1 0 0 1 0 1 1 1
                1 0 0 1 0 1 1 0 0 1 0 1 1 0 0 1 1 1
                1 0 0 0 1 1 1 0 0 1 0 1 0 1 1 0 1 1
                1 1 1 0 1 1 1 0 0 1 1 0 1 0 1 0 1 1
                1 1 0 1 1 1 1 0 0 1 1 1 0 1 0 0 1 1
                1 0 1 1 1 1 1 0 0 1 0 0 1 0 0 0 1 1
                1 0 1 1 0 0 1 0 0 1 0 0 1 1 1 1 0 1
                1 1 0 1 0 0 1 0 0 1 1 1 0 0 1 1 0 1
                1 1 1 0 0 0 1 0 0 1 1 0 1 1 0 1 0 1
                1 0 0 0 0 0 1 0 0 1 0 1 0 0 0 1 0 1
                1 0 0 1 1 0 1 0 0 1 0 1 1 1 1 0 0 1
                1 1 1 1 1 0 1 0 0 1 1 0 0 0 1 0 0 1
                1 1 0 0 1 0 1 0 0 1 1 1 1 1 0 0 0 1
                1 0 1 0 1 0 1 0 0 1 0 0 0 0 0 0 0 1
```

5. CRC-16(CCITT)

$G(x) = x^{16} + x^{12} + x^5 + 1$

burst lenth 17 : 1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 1

burst lenth 18 : 1 1 0 0 1 1 0 0 0 0 0 0 1 1 0 0 0 1 1

burst lenth 19 : 1 1 1 0 1 1 1 0 0 0 0 0 1 1 1 0 0 1 1 1

1 0 1 0 1 0 1 0 0 0 0 0 1 0 1 0 0 1 0 1

burst lenth 20 : 1 1 1 1 1 1 1 1 0 0 0 0 1 1 1 1 0 1 1 1 1

1 0 1 1 1 0 1 1 0 0 0 0 1 0 1 1 0 1 0 1 1

1 1 0 1 1 1 0 1 0 0 0 0 1 1 0 1 0 1 1 0 1

1 0 0 1 1 0 0 1 0 0 0 0 1 0 0 1 0 1 0 0 1

burst lenth 21 : 1 1 1 1 0 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1

1 0 1 1 0 0 1 1 1 0 0 0 1 0 1 1 1 1 0 1 1 1

1 1 0 1 0 1 0 1 1 0 0 0 1 1 0 1 1 1 1 0 1 1

1 0 0 1 0 0 0 1 1 0 0 0 1 0 0 1 1 1 0 0 1 1

1 1 1 0 0 1 1 0 1 0 0 0 1 1 1 0 1 1 1 1 0 1

1 0 1 0 0 0 1 0 1 0 0 0 1 0 1 0 1 1 0 1 0 1

1 1 0 0 0 1 0 0 1 0 0 0 1 1 0 0 1 1 1 0 0 1

1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 1 1 0 0 0 1

burst lenth 22 : 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 1 0 1 1 1 1 1

1 0 1 1 0 1 1 1 1 1 0 1 0 1 1 1 0 0 1 1 1 1

1 1 0 1 0 0 0 1 1 1 0 1 1 0 1 1 0 1 0 1 1 1

1 0 0 1 0 1 0 1 1 1 0 1 0 0 1 1 0 0 0 1 1 1

1 1 1 0 0 0 1 0 1 1 0 1 1 1 0 1 0 1 1 0 1 1

1 0 1 0 0 1 1 0 1 1 0 1 0 1 0 1 0 0 1 0 1 1

1 1 0 0 0 0 0 0 0 1 1 0 1 1 0 0 1 0 1 0 0 1 1

1 0 0 0 0 1 0 0 1 1 0 1 0 0 0 1 0 0 0 0 1 1

1 1 1 1 1 0 1 1 0 1 0 1 1 1 1 1 0 0 1 1 1 0 1

1 0 1 1 1 1 1 1 0 1 0 1 0 1 1 0 0 0 1 1 0 1

1 1 0 1 1 0 0 1 0 1 0 1 1 0 1 0 0 1 0 1 0 1

1 0 0 1 1 1 0 1 0 1 0 1 0 0 1 0 0 0 0 1 0 1

1 1 1 0 1 0 1 0 0 1 0 1 1 1 0 0 0 1 1 0 0 1

1 0 1 0 1 1 1 0 0 1 0 1 0 1 0 0 0 0 1 0 0 1

1 1 0 0 1 0 0 0 0 1 0 1 1 0 0 0 0 1 0 0 0 1

1 0 0 0 1 1 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 1

6. CRC-16(ANSI)

$G(x) = x^{16} + x^{15} + x^2 + 1$

burst lenth 17 : 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1

burst lenth 18 : 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1

burst lenth 19 : 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1

1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1

burst lenth 20 : 1 1 1 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 1

1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 1

1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 1 0 1 1 0 1

1 0 1 1 1 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 1

burst lenth 21 : 1 1 0 1 0 1 0 0 0 0 0 0 0 0 0 1 0 1 1 1 1 1

1 0 1 1 0 1 0 0 0 0 0 0 0 0 0 1 1 1 0 1 1 1

1 1 1 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 1 1

1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 1 0 0 0 1 1

1 0 1 0 1 1 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 1

```
                1 1 0 0 1 1 0 0 0 0 0 0 0 0 1 0 1 0 1 0 1
                1 0 0 1 1 1 0 0 0 0 0 0 0 0 1 1 0 1 0 0 1
                1 1 1 1 1 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1
burst lenth 22 : 1 0 1 0 1 0 1 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1
                1 1 0 0 1 0 1 0 0 0 0 0 0 0 1 0 1 0 1 1 1 1
                1 0 0 1 1 0 1 0 0 0 0 0 0 0 1 1 0 1 0 1 1 1
                1 1 1 1 1 0 1 0 0 0 0 0 0 0 1 0 0 0 0 1 1 1
                1 1 0 1 0 0 1 0 0 0 0 0 0 0 1 0 1 1 1 0 1 1
                1 0 1 1 0 0 1 0 0 0 0 0 0 0 1 1 1 0 1 0 1 1
                1 1 1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 1 0 0 1 1
                1 0 0 0 0 0 1 0 0 0 0 0 0 0 1 1 0 0 0 0 1 1
                1 0 0 1 0 1 1 0 0 0 0 0 0 0 1 1 0 1 1 1 0 1
                1 1 1 1 0 1 1 0 0 0 0 0 0 0 1 0 0 0 1 1 0 1
                1 0 1 0 0 1 1 0 0 0 0 0 0 0 1 1 1 1 0 1 0 1
                1 1 0 0 0 1 1 0 0 0 0 0 0 0 1 0 1 0 0 1 0 1
                1 1 1 0 1 1 1 0 0 0 0 0 0 0 1 0 0 1 1 0 0 1
                1 0 0 0 1 1 1 0 0 0 0 0 0 0 1 1 0 0 1 0 0 1
                1 1 0 1 1 1 1 0 0 0 0 0 0 0 1 0 1 1 0 0 0 1
                1 0 1 1 1 1 1 0 0 0 0 0 0 0 1 1 1 0 0 0 0 1
```

APPENDIX B: SIMULATION RESULTS USING BINOMIAL DISTRIBUTION

The following tables and figures are the simulation results for each CRC code using a binomial distribution random number generator. Each CRC code is listed in TABLE 2.2. Each table and figure shows the relationship between the burst error and block burst error for each CRC code. The source code for computing the simulation results is shown in APPENDIX D. 2.

TABLE B.1: SIMULATION RESULTS OF CRC-4(2048 BITS) USING BINOMIAL DISTRIBUTION

Total Burst Length	BER Statistical	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
15000	Mean	1572	203.9	1.811	1.006	1.000	1.000
	SD	140.7	284.3	7.610	0.271	0.003	0.000
	Mode($b > 0$)	1562	1	1	1	1	1
	Mode($b > 4$)	1562	5	5	5	5	0
20000	Mean	1697	232.2	2.424	1.003	1.000	1.000
	SD	140.7	311.9	11.93	0.127	0.009	0.001
	Mode($b > 0$)	1734	1	1	1	1	1
	Mode($b > 4$)	1734	5	5	5	5	0
25000	Mean	1629	213.0	1.741	1.003	1.000	1.000
	SD	140.7	295.0	7.713	0.173	0.015	0.000
	Mode($b > 0$)	1633	1	1	1	1	1
	Mode($b > 4$)	1633	5	5	5	5	0
300000	Mean	1745	226.7	1.655	1.001	1.000	1.000
	SD	140.7	311.6	7.921	0.101	0.003	0.000
	Mode($b > 0$)	1826	1	1	1	1	1
	Mode($b > 4$)	1826	19	5	5	5	5

TABLE B.2: SIMULATION RESULTS OF CRC-9(14280 BITS) USING BINOMIAL DISTRIBUTION

Total Burst Length	BER / Statistical	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
15000	Mean	7185	4796	306.2	6.488	1.020	1.000
	SD	140.7	1360	664.0	42.98	0.594	0.000
	Mode($b > 0$)	7337	1	1	1	1	1
	Mode($b > 9$)	7337	4385	10	10	10	0
20000	Mean	9693	7094	431.1	4.627	1.117	1.008
	SD	140.7	1403	853.9	28.75	2.530	0.551
	Mode($b > 0$)	9837	1	1	1	1	1
	Mode($b > 9$)	9837	6006	10	10	10	10
25000	Mean	12195	9527	660.3	7.822	1.149	1.000
	SD	140.7	1412	1185	54.73	4.052	0.000
	Mode($b > 0$)	12337	1	1	1	1	1
	Mode($b > 9$)	12337	7909	10	10	10	10
300000	Mean	13339	10691	696.2	2.832	1.008	1.000
	SD	140.7	1413	1369	27.58	0.641	0.025
	Mode($b > 0$)	13428	10975	1	1	1	1
	Mode($b > 9$)	13428	10975	10	10	10	10

TABLE B.3: SIMULATION RESULTS OF CRC-12(1880 BITS) USING BINOMIAL DISTRIBUTION

Total Burst Length	BER / Statistical	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
15000	Mean	1580	191.7	1.706	1.006	1.000	1.000
	SD	140.7	273.9	7.012	0.271	0.006	0.000
	Mode($b > 0$)	1591	1	1	1	1	1
	Mode($b > 12$)	1591	13	13	13	13	0
20000	Mean	1518	168.3	1.886	1.001	1.000	1.000
	SD	140.7	250.8	8.391	0.070	0.007	0.003
	Mode($b > 0$)	1530	1	1	1	1	1
	Mode($b > 12$)	1530	13	13	13	13	13
25000	Mean	1489	166.2	1.516	1.002	1.000	1.000
	SD	140.7	249.4	5.850	0.125	0.015	0.000
	Mode($b > 0$)	1523	1	1	1	1	1
	Mode($b > 12$)	1523	13	13	13	13	0
300000	Mean	1577	176.1	1.483	1.001	1.000	1.000
	SD	140.7	262.6	6.276	0.081	0.002	0.000
	Mode($b > 0$)	1659	1	1	1	1	1
	Mode($b > 12$)	1659	13	13	13	13	0

TABLE B.4: SIMULATION RESULTS OF CRC-16(256 BITS) USING BINOMIAL DISTRIBUTION

Total Burst Length	BER / Statistical	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
15000	Mean	42.81	1.169	1.004	1.000	1.000	1.000
	SD	46.39	1.596	0.025	0.001	0.000	0.000
	Mode($b > 0$)	1	1	1	1	1	1
	Mode($b > 16$)	32	17	0	0	0	0
20000	Mean	42.58	1.171	1.000	1.000	1.000	1.000
	SD	46.27	1.615	0.027	0.000	0.000	0.000
	Mode($b > 0$)	1	1	1	1	1	1
	Mode($b > 16$)	43	17	0	0	0	0
25000	Mean	44.17	1.174	1.000	1.000	1.000	1.000
	SD	47.38	1.646	0.021	0.001	0.000	0.000
	Mode($b > 0$)	1	1	1	1	1	1
	Mode($b > 16$)	33	17	0	0	0	0
300000	Mean	44.42	1.155	1.000	1.000	1.000	0.000
	SD	47.61	1.532	0.021	0.000	0.000	0.000
	Mode($b > 0$)	1	1	1	1	1	1
	Mode($b > 16$)	60	17	0	0	0	0

TABLE B.5: SIMULATION RESULTS OF CRC-16(1024 BITS) USING BINOMIAL DISTRIBUTION

Total Burst Length	BER / Statistical	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
15000	Mean	700.1	27.08	1.058	1.000	1.000	1.000
	SD	139.7	66.95	1.072	0.035	0.001	0.000
	Mode($b > 0$)	721	1	1	1	1	1
	Mode($b > 16$)	721	17	17	17	17	0
20000	Mean	702.8	25.89	1.092	1.000	1.000	1.000
	SD	139.8	64.98	1.530	0.017	0.001	0.000
	Mode($b > 0$)	741	1	1	1	1	1
	Mode($b > 16$)	741	17	17	17	17	17
25000	Mean	706.2	26.22	1.055	1.000	1.000	1.000
	SD	139.8	65.76	1.095	0.032	0.002	0.000
	Mode($b > 0$)	739	1	1	1	1	1
	Mode($b > 16$)	739	17	17	17	17	0
300000	Mean	727.7	25.73	1.045	1.000	1.000	1.000
	SD	139.9	65.78	1.073	0.013	0.000	0.000
	Mode($b > 0$)	813	1	1	1	1	1
	Mode($b > 16$)	813	17	17	0	0	0

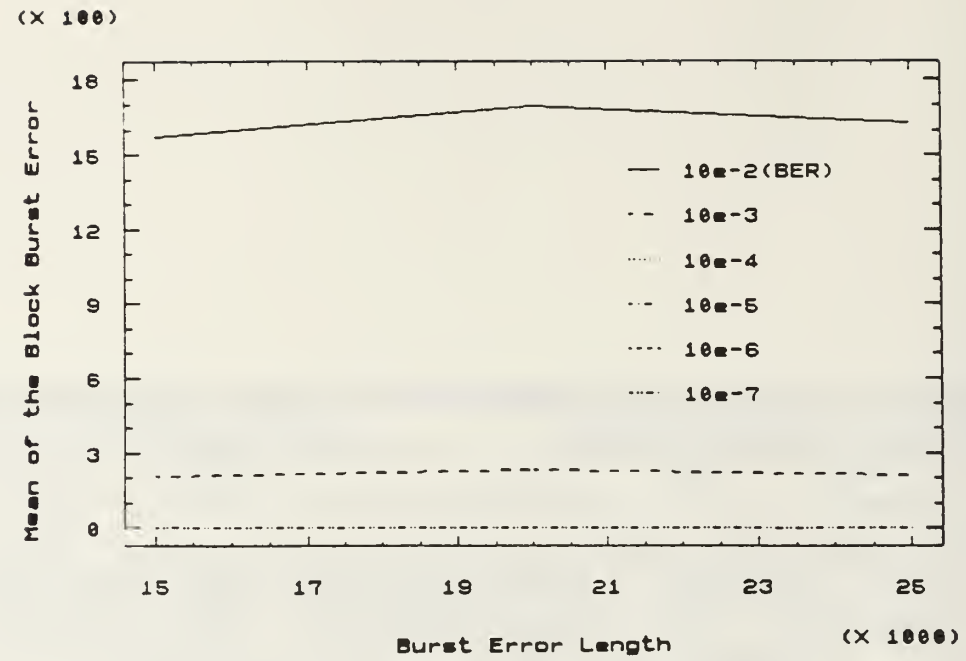


Figure B.1: Distribution of CRC-4(2048 Bits)

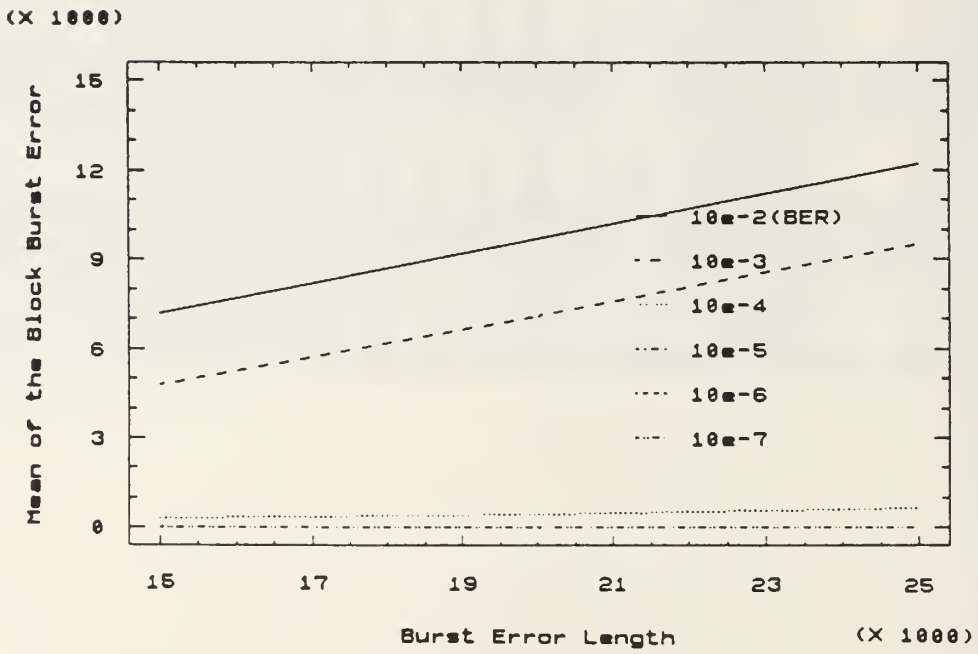


Figure B.2: Distribution of CRC-9(14280 Bits)

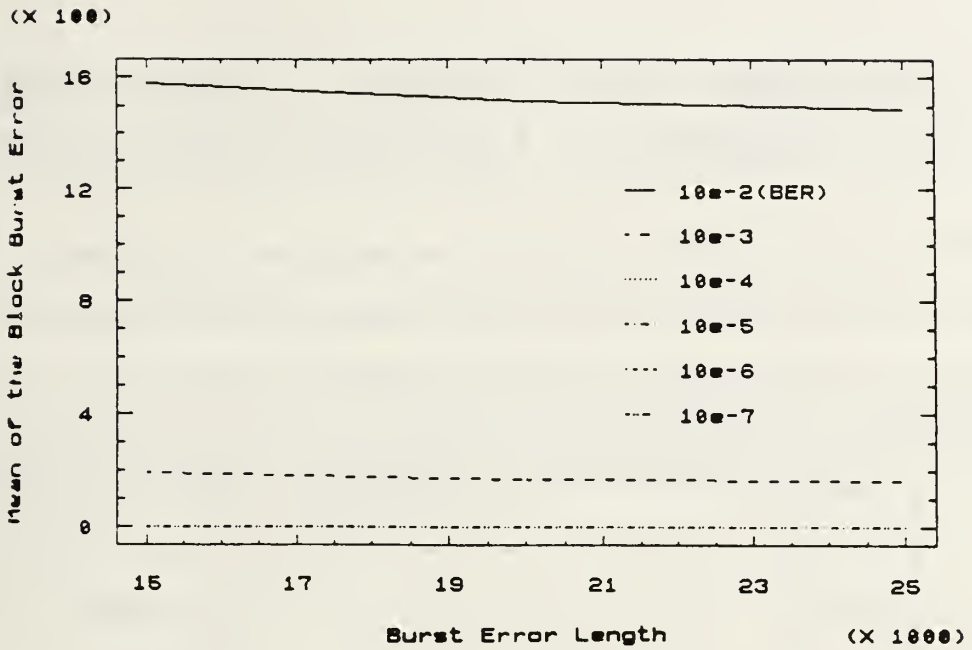


Figure B.3: Distribution of CRC-12(1880 Bits)

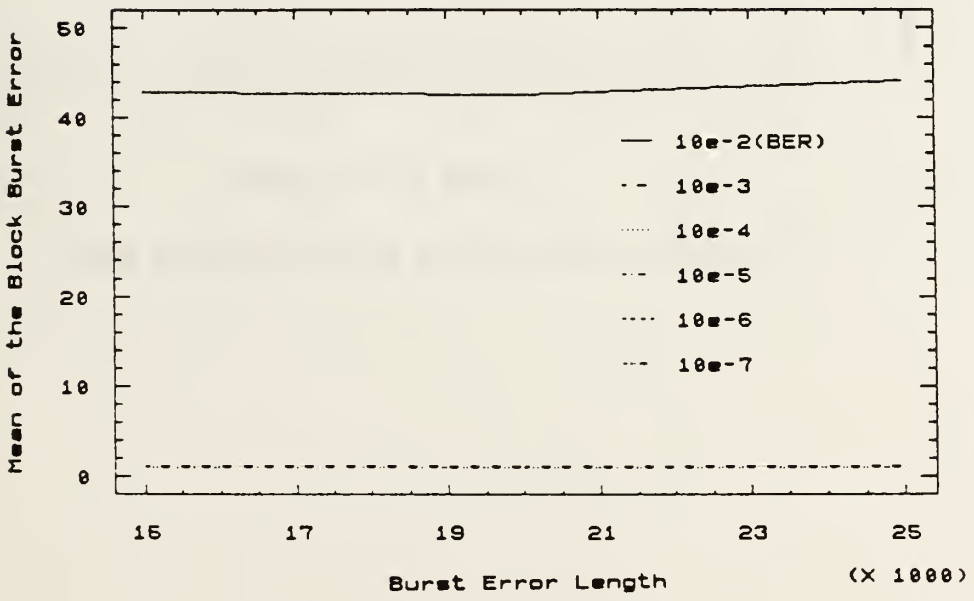


Figure B.4: Distribution of CRC-16(256 Bits)

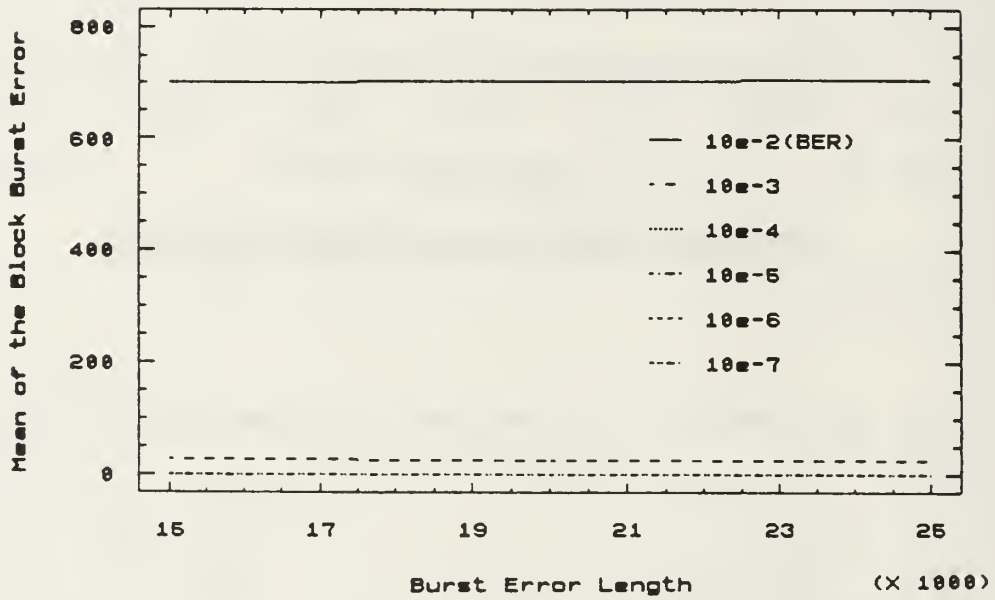


Figure B.5: Distribution of CRC-16(1024 Bits)

APPENDIX C: SIMULATION RESULTS USING EXPONENTIAL DISTRIBUTION

The following tables and figures are the simulation results for each CRC code using a exponential distribution random number generator. Each CRC code is listed in TABLE 2.2. Each table and figure shows the relationship between the burst error and block burst error for each CRC code. The source code for computing the simulation results is shown in APPENDIX D. 3.

TABLE C.1: SIMULATION RESULTS OF CRC-4(2048 BITS) USING EXPONENTIAL DISTRIBUTION

Total Burst Length	BER / Statistical	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
15000	Mean	1578	188.6	1.966	1.011	1.000	1.000
	SD	140.7	269.7	8.877	0.421	0.000	0.000
	Mode($b > 0$)	1533	1	1	1	1	1
	Mode($b > 4$)	1533	5	5	5	0	0
20000	Mean	1698	214.2	1.917	1.005	1.000	1.000
	SD	140.7	296.1	9.235	0.210	0.000	0.001
	Mode($b > 0$)	1695	1	1	1	1	1
	Mode($b > 4$)	1695	5	5	5	0	0
25000	Mean	1622	207.8	1.898	1.004	1.000	1.000
	SD	140.7	290.1	8.916	0.019	0.001	0.000
	Mode($b > 0$)	1645	1	1	1	1	1
	Mode($b > 4$)	1645	5	5	5	5	0
300000	Mean	1743	222.4	1.676	1.001	1.000	1.000
	SD	140.7	307.6	8.108	0.111	0.003	0.000
	Mode($b > 0$)	1822	1	1	1	1	1
	Mode($b > 4$)	1822	33	5	5	5	5

TABLE C.2: SIMULATION RESULTS OF CRC-9(14280 BITS) USING EXPONENTIAL DISTRIBUTION

Total Burst Length	BER / Statistical	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
15000	Mean	7209	4732	286.0	10.10	1.000	1.000
	SD	140.7	1354	623.3	66.76	0.000	0.000
	Mode($b > 0$)	7338	1	1	1	1	1
	Mode($b > 9$)	7338	2203	10	10	0	0
20000	Mean	9709	6922	416.4	4.425	1.000	1.000
	SD	140.7	1400	852.1	30.05	0.000	0.000
	Mode($b > 0$)	9838	1	1	1	1	1
	Mode($b > 9$)	9838	5194	10	10	0	0
25000	Mean	12211	9494	616.9	8.400	1.014	1.000
	SD	140.7	1412	1129	52.31	0.446	0.000
	Mode($b > 0$)	12338	1	1	1	1	1
	Mode($b > 9$)	12338	8151	10	10	10	0
300000	Mean	13334	10644	722.8	3.207	1.007	1.000
	SD	140.7	1413	1407	32.15	0.546	0.014
	Mode($b > 0$)	13443	11045	1	1	1	1
	Mode($b > 9$)	13443	11045	10	10	10	10

TABLE C.3: SIMULATION RESULTS OF CRC-12(1880 BITS) USING EXPONENTIAL DISTRIBUTION

Total Burst Length	BER / Statistical	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
15000	Mean	1573	183.6	1.926	1.007	1.000	1.000
	SD	140.7	264.5	8.767	0.292	0.000	0.000
	Mode($b > 0$)	1565	1	1	1	1	1
	Mode($b > 12$)	1565	13	13	13	0	0
20000	Mean	1518	157.1	1.694	1.004	1.000	1.000
	SD	140.7	239.6	7.268	0.188	0.000	0.000
	Mode($b > 0$)	1538	1	1	1	1	1
	Mode($b > 12$)	1538	13	13	13	0	0
25000	Mean	1489	163.3	1.671	1.003	1.000	1.000
	SD	140.7	246.2	7.068	0.140	0.001	0.000
	Mode($b > 0$)	1502	1	1	1	1	1
	Mode($b > 12$)	1502	13	13	13	13	0
300000	Mean	1578	172.6	1.491	1.001	1.000	1.000
	SD	140.7	259.0	6.379	0.087	0.002	0.000
	Mode($b > 0$)	1656	1	1	1	1	1
	Mode($b > 12$)	1656	13	13	13	13	0

TABLE C.4: SIMULATION RESULTS OF CRC-16(256 BITS) USING EXPONENTIAL DISTRIBUTION

Total Burst Length	BER / Statistical	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
15000	Mean	43.60	1.164	1.004	1.000	1.000	1.000
	SD	46.96	1.554	0.026	0.001	0.000	0.000
	Mode($b > 0$)	1	1	1	1	1	1
	Mode($b > 16$)	35	17	0	0	0	0
20000	Mean	42.92	1.155	1.000	1.000	1.000	1.000
	SD	46.49	1.500	0.025	0.001	0.000	0.000
	Mode($b > 0$)	1	1	1	1	1	1
	Mode($b > 16$)	39	17	0	0	0	0
25000	Mean	43.52	1.155	1.000	1.000	1.000	1.000
	SD	46.92	1.508	0.023	0.001	0.000	0.000
	Mode($b > 0$)	1	1	1	1	1	1
	Mode($b > 16$)	45	17	0	0	0	0
300000	Mean	43.86	1.155	1.000	1.000	1.000	0.000
	SD	47.23	1.530	0.021	0.000	0.000	0.000
	Mode($b > 0$)	1	1	1	1	1	1
	Mode($b > 16$)	59	17	0	0	0	0

TABLE C.5: SIMULATION RESULTS OF CRC-16(1024 BITS) USING EXPONENTIAL DISTRIBUTION

Total Burst Length	BER / Statistical	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
15000	Mean	701.5	25.32	1.079	1.001	1.000	1.000
	SD	139.7	63.42	1.360	0.050	0.001	0.000
	Mode($b > 0$)	1	1	1	1	1	1
	Mode($b > 16$)	661	17	17	17	0	0
20000	Mean	701.2	23.45	1.067	1.000	1.000	1.000
	SD	139.7	60.28	1.266	0.024	0.000	0.000
	Mode($b > 0$)	733	1	1	1	1	1
	Mode($b > 16$)	733	17	17	17	0	0
25000	Mean	699.8	24.22	1.065	1.000	1.000	1.000
	SD	139.7	61.79	1.234	0.026	0.001	0.000
	Mode($b > 0$)	745	1	1	1	1	1
	Mode($b > 16$)	745	17	17	17	17	0
300000	Mean	725.4	25.18	1.048	1.000	1.000	1.000
	SD	140.0	64.73	1.110	0.015	0.000	0.000
	Mode($b > 0$)	812	1	1	1	1	1
	Mode($b > 16$)	812	17	17	17	0	0

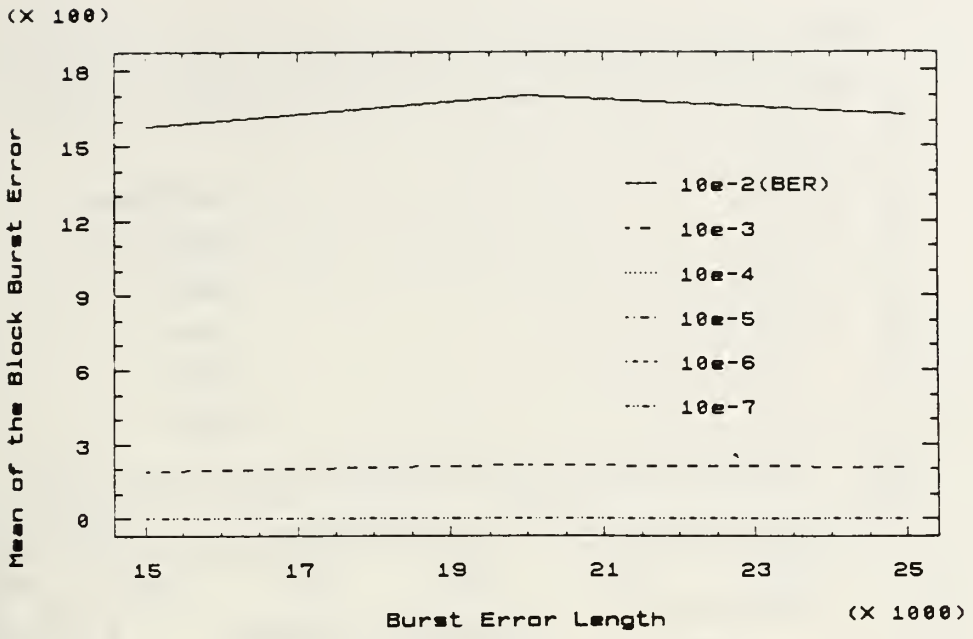


Figure C.1: Distribution of CRC-4(2048 Bits)

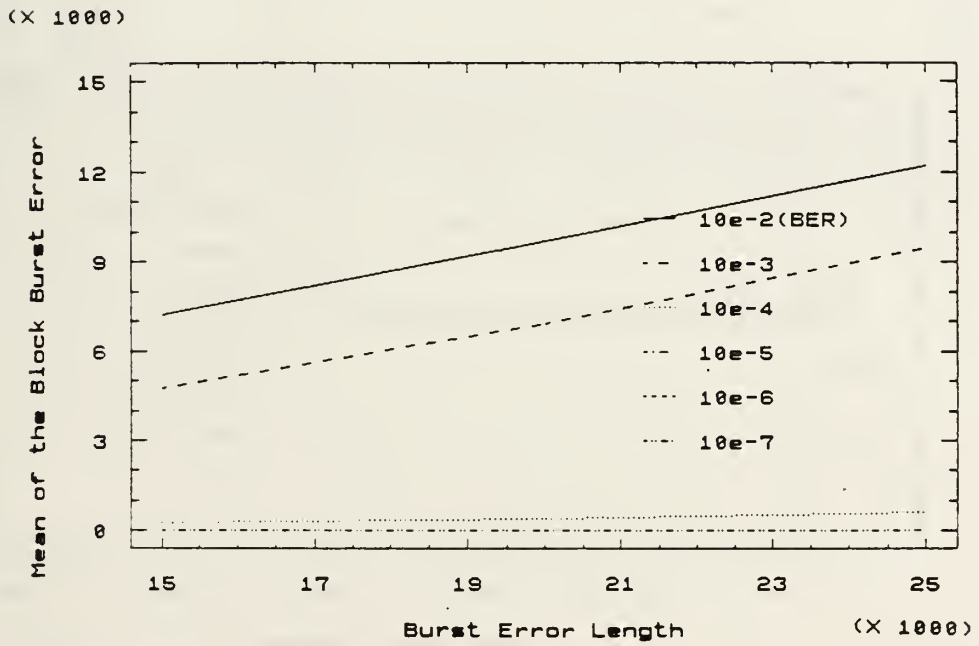


Figure C.2: Distribution of CRC-9(14280 Bits)

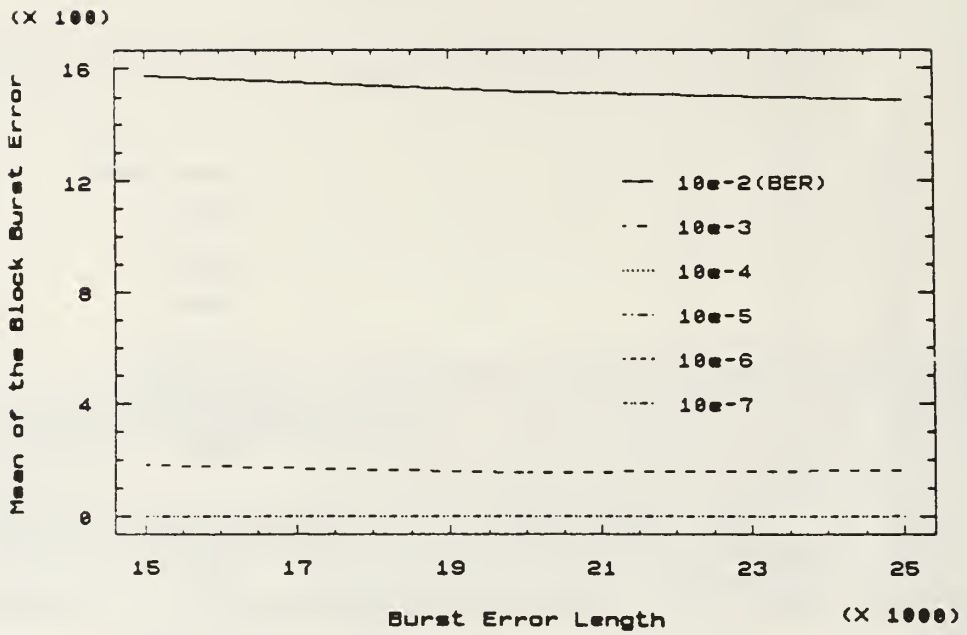


Figure C.3: Distribution of CRC-12(1880 Bits)

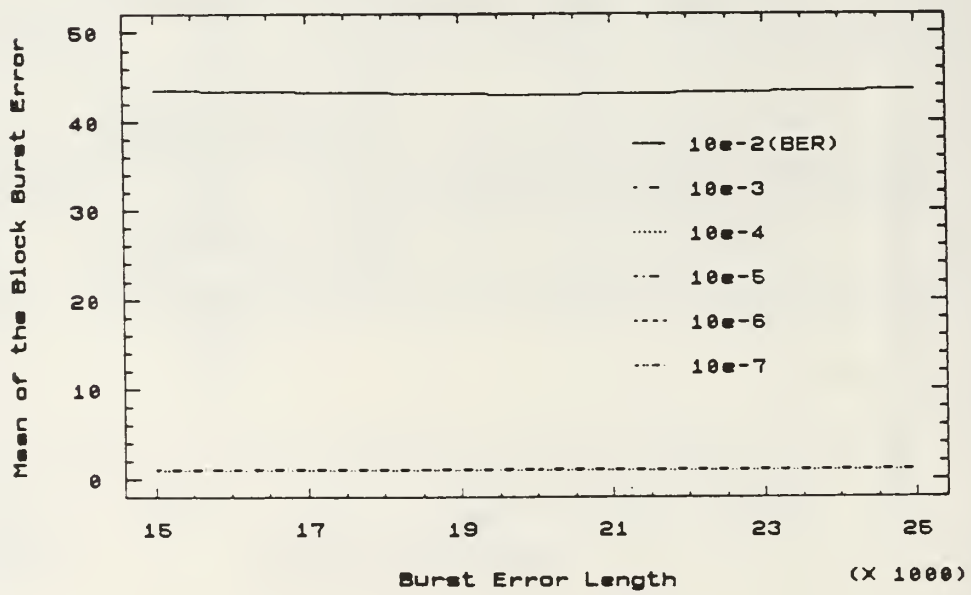


Figure C.4: Distribution of CRC-16(256 Bits)

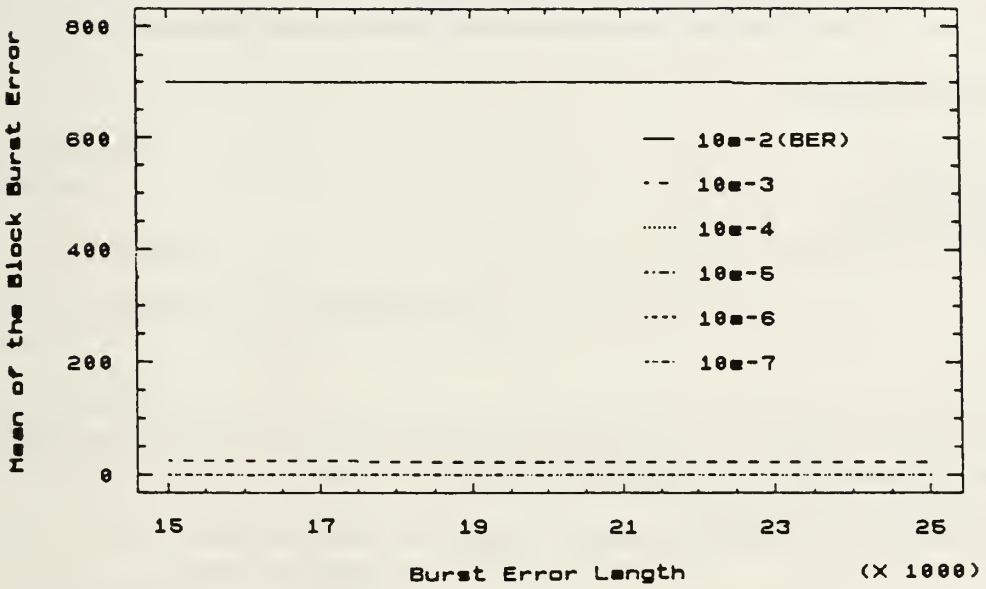


Figure C.5: Distribution of CRC-16(1024 Bits)

APPENDIX D: PROGRAM LISTS

1. Undetected Burst Error Bit Pattern Generator

```
/*********************************************************************
 * Source       : pattern.c                                          *
 * Author       : Yoon, Hee Byung                                    *
 * Date         : AUG  19 1990                                       *
 * Update       : NOV  24 1990                                       *
 * Description  : To find out the undetected burst error bit         *
 *                patterns for each CRC codes.                       *
 *********************************************************************/

#include <stdio.h>

int rn[20];
int keep[20];
int restore[20];
int Gx[7] ={1,0,0,0,0,1,1};

main()
{
int an,i;
int L;
printf("\nBit Pattern for the undetected burst error (CRC-6)\n");
printf("G(x) = x6 + x + 1\n\n");
/* for burst error 7 to 10
  for burst len 7: generate length 5 sequence 00000 to 11111
                8                  6          000000 to 111111
                9                  7          0000000 to 1111111
                10                 8          00000000 to 11111111
                   .......
                 etc.
*/
for(L=5; L<= 11; L++) {
                        printf("\nburst lenth %d \n", L+2);
                        rn[0]= rn[L+1]=1;
                        generate(L, L);
                      }
}


generate(l, L)
int l, L;
{
int i;
```

```
if(l==1) { rn[l]=1;
           for(i=0; i<=L+1; i++) restore[i]=rn[i];
           divide(L+2);
           for(i=0; i<=L+1; i++) rn[i]=restore[i];

           rn[l]=0;

           for(i=0; i<=L+1; i++) restore[i]=rn[i];
           divide(L+2);
           for(i=0; i<=L+1; i++) rn[i]=restore[i];
           return;
          }

rn[l]=1;
generate(l-1, L);
rn[l]=0;
generate(l-1, L);

}

divide(burstlen)
int burstlen;
{
int i, j, k, current;

for(i=0; i<=burstlen-1; i++) keep[i]=rn[i];
/* divide */

for(i=0; i<=burstlen-7; ) {
      current=i;
      for(j=current; j< current+7; j++) rn[j] = (rn[j] == Gx[j-current])? 0: 1;

      while (rn[current++]==0);

      i = current -1;
     }


if(current > burstlen) {
for(i=0; i<=burstlen-1; i++) printf("%d ",keep[i]);
 printf("\n");

        }
}
```

2. Binomial Distribution

```
/*********************************************************************
 * Source       : bcrc.c                                             *
```

```
 * Auther       : Yoon, Hee Byung                                        *
 * Date         : OCT 5 1990                                             *
 * Update       : OCT 27 1990                                            *
 * Description  : To find out the result(mean, mode, SD) for the         *
 *                binomial distribution randon number with BER.          *
 ************************************************************************/

#include <stdio.h>
#include <math.h>

#define Ntimes  100
#define MAX_BUF 25000
#define Bur_Len 2048
#define Blocks  13
#define Deg_Gx  4
#define SEED    12345

double e;
double ber[6] ={0.01, 0.001, 0.0001, 0.00001, 0.000001, 0.0000001};
int rn[MAX_BUF];

main(argc, argv)
int argc;
char **argv;
{
int  i,j,m,n,k,l;
int  cnt1,cnt2,cnt3;
int  bur_len,new_len;
int  ie, ntimes,max_buf;
int     mode1,mode3;
double  r,x,x1,y,error;
double  mean[200],mean1,mean2,mean3;
double  var[200],sd[200],sd1,sd2;
double  mode2,mode4;
        void    randnum();

for(ie=0; ie<=5; ie++) { e = ber[ie];
        mean1   = mean3 = sd1 = sd2 = 0.0;
  error   = e;
  max_buf = MAX_BUF;
ntimes  = Ntimes;
mode1 = mode3 = 0;
mode2 = mode4 = 0.0;

/*  cnt1: for detecting 1st bit error position
    cnt2:     last bit position
    cnt3:     computing average burst length
*/
for (k=1; k <= Ntimes; k++) {
    randnum();
    cnt1 = cnt2 = cnt3 = 0;
```

```
      new_len = bur_len = 0;
    mean2 = x1 = x = y = 0.0;
    mean[k] = var[k] = sd[k] = 0.0;

            for (n=1; n <= Blocks; n++) {
                cnt2=cnt1=0;

                if (n == Blocks) {
              for (m=(n-1)*Bur_Len; m <= MAX_BUF-1; m++) {
      if (rn[m] == 1) {
         cnt1++;
         break; }
      else cnt1++;
           }

   for (m=MAX_BUF-1; m >= (n-1)*Bur_Len; m--) {
         if (rn[m] ==1) {
          cnt2++;
          break;  }
       else cnt2++;
   }
new_len = MAX_BUF-(n-1)*Bur_Len - cnt1 - cnt2 + 2;
        }

                else {    /* not the last block */
          for (m=0; m <= n*Bur_Len-1; m++) {
                      cnt1=cnt2=0;
      if (rn[m] == 1) {
         cnt1++;
         break;}
       else cnt1++;
             }

          for (m=n*Bur_Len-1; m >= Bur_Len*(n-1); m--) {
         if (rn[m] ==1) {
          cnt2++;
          break; }
        else cnt2++;
             }

   new_len = Bur_Len - cnt1 - cnt2 + 2;
               }
               bur_len = bur_len +new_len;
        cnt3++;
            }

           bur_len = bur_len/cnt3;
    printf("\n total bur_len = %d\n",bur_len);

/* Compute mean and standard deviation with burst length */
           x = x1 = y = 0.0;
```

```
    x1 = (bur_len*e*pow(1-e,(double)(bur_len-1)))/
                (1-pow(1-e,(double)bur_len));


    for(i=2; i<=bur_len; i++) {
                r = 0.0;
                r = ((bur_len+1-i)*pow(e, (double)2)*
                      pow(1-e,(double)(bur_len-i)))/
                    (1-pow(1-e,(double) bur_len));
       x = x+(i*r);
                y = y+pow((double)i,(double)2)*r;
       if (i > Deg_Gx) {
   if (mode4 < r) {
       mode4 = r;
       mode3 = i;
   }
                }
       if (mode2 < r) {
   mode2 = r;
           mode1 = i;
       }
      }
    if (mode2 < x1) {
mode2 = x1;
mode1 = 1;
    }

    mean[k]  = x+x1;
    mean1    = mean1 + mean[k];
    mean2    = pow((double) (mean[k]),(double)2);
    var[k]   = (x1+y)-mean2;
           sd[k]    = sqrt(var[k]);
    sd1      = sd1 +sd[k];
    mean[k] = sd[k] = var[k] = 0.0;

}
mean3 = mean1/Ntimes;
        sd2 = sd1/Ntimes;
        printf("\nTotal CRC Block Size = %d\n",max_buf);
        printf("\nBit Error Ratio      = %e\n",ber[ie]);
printf("\nNew Mean (%d times simulation) = %e\n",ntimes,mean3);
printf("\nNew SD (%d times simulation)   = %e\n",ntimes,sd2);
printf("\nWhen Ber_Len >= 1   ==> Mode = %d\n",mode1);
printf("\nWhen Ber_Len >= %d  ==> Mode = %d\n",Deg_Gx,mode3);
    }
}

void randnum()
{
int j;
float RN,RN1;
```

```
for (j=1; j<=MAX_BUF-2; j++) {
            RN = rand();
            RN1 = RN/2147483648.0;
    if (RN1 <= e)
             rn[j] = 1;
            else
             rn[j] = 0;
        }
        rn[0] = 1;
        rn[MAX_BUF-1] = 1;
        return;
 }
```

3. Exponential Distribution

```
/*********************************************************************
 * Source       : ecrc.c                                             *
 * Author       : Yoon, Hee Byung                                    *
 * Date         : NOV 12 1990                                        *
 * Update       : DEC  7 1990                                        *
 * Description  : To find out the result(mean, mode, SD) for         *
 *                the exponential distribution with BER.             *
 *********************************************************************/

#include <stdio.h>
#include <math.h>

#define Ntimes  100
#define MAX_BUF 25000
#define Bur_Len 1024
#define Blocks  25
#define Deg_Gx  16
#define SEED    12345

double e;
double ber[6] ={0.01, 0.001, 0.0001, 0.00001, 0.000001, 0.0000001};
int rn[MAX_BUF];

main(argc, argv)
int argc;
char **argv;
{
int  i,j,m,n,k,l;
int  cnt1,cnt2,cnt3;
int  bur_len,new_len;
int  ie, ntimes,max_buf;
int     mode1,mode3;
double  r,x,x1,y,error;
double  mean[200],mean1,mean2,mean3;
```

```
double  var[200],sd[200],sd1,sd2;
double  mode2,mode4;
        void    randnum();

for(ie=0; ie<=5; ie++) { e = ber[ie];
        mean1   = mean3 = sd1 = sd2 = 0.0;
  error   = e;
  max_buf = MAX_BUF;
ntimes  = Ntimes;
mode1 = mode3 = 0;
mode2 = mode4 = 0.0;

/*  cnt1: for detecting 1st bit error position
    cnt2:     last bit position
    cnt3:     computing average burst length
*/

for (k=1; k <= Ntimes; k++) {
    randnum();
    cnt1 = cnt2 = cnt3 = 0;
      new_len = bur_len = 0;
    mean2 = x1 = x = y = 0.0;
    mean[k] = var[k] = sd[k] = 0.0;

            for (n=1; n <= Blocks; n++) {
                cnt2=cnt1=0;

                if (n == Blocks) {
           for (m=(n-1)*Bur_Len; m <= MAX_BUF-1; m++) {
      if (rn[m] == 1) {
         cnt1++;
         break; }
      else cnt1++;
           }

   for (m=MAX_BUF-1; m >= (n-1)*Bur_Len; m--) {
         if (rn[m] ==1) {
          cnt2++;
          break;  }
       else cnt2++;
   }
new_len = MAX_BUF-(n-1)*Bur_Len - cnt1 - cnt2 + 2;
        }

                else {    /* not the last block */
           for (m=0; m <= n*Bur_Len-1; m++) {
                    cnt1=cnt2=0;
      if (rn[m] == 1) {
         cnt1++;
         break;}
       else cnt1++;
```

```
            }

          for (m=n*Bur_Len-1; m >= Bur_Len*(n-1); m--) {
        if (rn[m] ==1) {
         cnt2++;
         break; }
       else cnt2++;
            }

   new_len = Bur_Len - cnt1 - cnt2 + 2;
              }
              bur_len = bur_len +new_len;
       cnt3++;
           }

          bur_len = bur_len/cnt3;
    printf("\n total bur_len = %d\n",bur_len);

/* Compute mean and standard deviation with burst length */
          x = x1 = y = 0.0;
    x1 = (bur_len*e*pow(1-e,(double)(bur_len-1)))/
              (1-pow(1-e,(double)bur_len));

    for(i=2; i<=bur_len; i++) {
            r = 0.0;
            r = ((bur_len+1-i)*pow(e, (double)2)*
                  pow(1-e,(double)(bur_len-i)))/
                (1-pow(1-e,(double) bur_len));
       x = x+(i*r);
            y = y+pow((double)i,(double)2)*r;
       if (i > Deg_Gx) {
   if (mode4 < r) {
       mode4 = r;
       mode3 = i;
   }
            }
       if (mode2 < r) {
   mode2 = r;
         mode1 = i;
       }
      }
    if (mode2 < x1) {
mode2 = x1;
mode1 = 1;
    }
    mean[k]  = x+x1;
    mean1    = mean1 + mean[k];
    mean2    = pow((double) (mean[k]),(double)2);
    var[k]   = (x1+y)-mean2;
          sd[k]    = sqrt(var[k]);
    sd1      = sd1 +sd[k];
```

```
    mean[k] = sd[k] = var[k] = 0.0;
}
mean3 = mean1/Ntimes;
        sd2 = sd1/Ntimes;
        printf("\nTotal CRC Block Size = %d\n",max_buf);
        printf("\nBit Error Ratio      = %e\n",ber[ie]);
printf("\nNew Mean (%d times simulation) = %e\n",ntimes,mean3);
printf("\nNew SD (%d times simulation)   = %e\n",ntimes,sd2);
printf("\nWhen Ber_Len >= 1   ==> Mode = %d\n",mode1);
printf("\nWhen Ber_Len >= %d  ==> Mode = %d\n",Deg_Gx,mode3);
    }
}

void randnum()
{
int i,j,x;
float RN,RN1;
        for (i=0; i<=MAX_BUF-1; i++) {
            rn[i] = 0;
        }
for (j=1; j<=MAX_BUF-2; ) {
            RN = rand();
            RN1 = RN/2147483648.0;
    x = ceil(-(1.0/e)*(log(RN1)));
    j = j + x;
    if (j <= MAX_BUF-2) {
                rn[j] = 1;
                x = 0;
            }
            else {
                break;
    }
        }
        rn[0] = 1;
        rn[MAX_BUF-1] = 1;
        return;
 }
```

REFERENCES

1. Jai H. EU, "An Evaluation of Error Performance Estimation Schemes for DS1 Transmission Systems Carrying Live Traffic," *IEEE Trans. on Communications*, Vol. 30, No. 3, pp. 384-391, March 1990

2. "Extended Superframe Format(ESF) Interface Specification," *Bell Commun. Res.*, TR-TSY-000194, Issue 1, December 1987

3. E. N. Gilbert, "Capacity of a Burst-Noise Channel," *Bell Syst. Tech. J.*, Vol. 39, PT. 2, pp. 1253-1265, September 1960

4. R. E. Mallon and S. Ravikumar, "Detection of Burst Error Conditions Through Analysis of Performance Information," *GLOBECOM Rec.*, Vol. 3, pp. 2020-2024, 1987

5. C. B. Schlegel and M. A. Herro, "A Burst-Error-Correcting Viterbi Algorithm," *IEEE Trans. on Communications*, Vol. 38, No. 3, pp. 285-291, March 1990

6. E. O. Elliott, "Estimates of Error Rates for Codes on Burst-Noise Channels," *Bell Syst. Tech. J.*, pp. 1977-1997, September 1963

7. Andrew S. Tanenbaum, *Computer Networks*, pp. 204-212, Prentice-Hall, New Jersey, 1988

8. W. Stallings, *Data and Computer Communications*, pp. 107-112, Macmillan, New York, 1988

9. W. W. Rollins, "Error-Second Measurements as Performance Indicators for Digital Transmission Systems," *Telecommunication*, pp. 80. 82. 132, September 1980

10. K. A. Witzke and C. Leung, "A Comparison of Some Error Detecting CRC Code Standards," *IEEE Trans. on Communications*, Vol. COM-33, pp. 996-998, September 1985

11. Shu Lin and D. J. Costello, Jr., *Error Control Coding Fundamentals and Applications*, pp. 111-116, Prentice-Hall, New Jersey, 1983

12. Alberto Leon-Garcia, *Probability and Random Processes for Electrical Engineering*, pp. 109-131, Addison-Wesley, 1989

13. Averill M. Law and W. David Kelton, *Simulation Modeling and Analysis*, pp. 242-247, McGraw-Hill, New York, 1982

INITIAL DISTRIBUTION LIST

		No. of Copies
1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Library, Code 52 Naval Postgraduate School Monterey, CA 93943-5002	2
3.	Chairman, Code EC Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943-5002	1
4.	Prof. Chyan Yang, Code EC/Ya Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943-5002	2
5.	Prof. Tri T. Ha, Code EC/Ha Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943-5002	1
6.	Dr. Jai H. Eu 812 Thatcher Way Raleigh, NC 27615	2
7.	Library of the Naval Acedemy Angok dong, Jinhae city, Gyungnam 602-00 Republic of Korea	1
8.	Yoon, Hee Byung 32-7 Weulgae 1 dong, Nowon ku, Seoul Republic of Korea	4

Thesis
Y526 Yoon
c.1 The error performance analysis over cyclic redundancy check codes.